ENHANCEMENT OF BEES ALGORITHM FOR GLOBAL OPTIMISATION

by

MUHAMMAD SYAHRIL BAHARI

A thesis submitted to the

University of Birmingham

for the degree of

DOCTOR OF PHILOSOPHY

Department of Mechanical Engineering

School of Engineering

College of Engineering and Physical Sciences

University of Birmingham

September 2018

ABSTRACT

This research focuses on the improvement of the Bees Algorithm, a swarm-based nature-inspired optimisation algorithm that mimics the foraging behaviour of honeybees. The algorithm consists of exploitation and exploration, the two key elements of optimisation techniques that help to find the global optimum in optimisation problems. This thesis presents three new approaches to the Bees Algorithm in a pursuit to improve its convergence speed and accuracy.

The first proposed algorithm focuses on intensifying the local search area by incorporating Hooke and Jeeves' method in its exploitation mechanism. This direct search method contains a pattern move that works well in the new variant named "Bees Algorithm with Hooke and Jeeves" (BA-HJ). The second proposed algorithm replaces the randomly generated recruited bees deployment method with chaotic sequences using a well-known logistic map. This new variant called "Bees Algorithm with Chaos" (ChaosBA) was intended to use the characteristic of chaotic sequences to escape from local optima and at the same time maintain the diversity of the population. The third improvement uses the information of the current best solutions to create new candidate solutions probabilistically using the Estimation Distribution Algorithm (EDA) approach. This new version is called Bees Algorithm with Estimation Distribution (BAED).

Simulation results show that these proposed algorithms perform better than the standard BA, SPSO2011 and qABC in terms of convergence for the majority of the tested benchmark functions. The BA-HJ outperformed the standard BA in thirteen out of fifteen benchmark functions and is more effective in eleven out of fifteen benchmark functions when compared to SPSO2011 and qABC. In the case of the ChaosBA, the algorithm outperformed the standard BA in twelve out of fifteen benchmark functions and significantly better in eleven out of fifteen

test functions compared to qABC and SPSO2011. BAED discovered the optimal solution with the least number of evaluations in fourteen out of fifteen cases compared to the standard BA, and eleven out of fifteen functions compared to SPSO2011 and qABC. Furthermore, the results on a set of constrained mechanical design problems also show that the performance of the proposed algorithms is comparable to those of the standard BA and other swarm-based algorithms from the literature.

ACKNOWLEDGEMENTS

First and foremost, I thank Allah for His greatness and mercifulness for giving me the strength and courage to complete this thesis. These outstanding individuals have done so much to make this possible:

My sincere thanks to my supervisor, Professor Duc Truong Pham for his acceptance, guidance, knowledge and countless supervision throughout my study. His continuous encouragement has given me the motivation to complete this research.

My special thanks to the Government of Malaysia through the Ministry of Education and Universiti Malaysia Perlis for the sponsorship during the tenure of my study.

My utmost appreciation to my research colleagues, Dr. Shafie, Dr. Silah Hayati, Dr. Nik Mohd Farid, Al-Anthoni Akhmad and Turki Al-Bakir for their continuous supports and encouragements.

My deepest gratitude to my parents, in-laws and family members for their love, supports and countless prayers.

My heartfelt gratitude to my beloved wife, Dr. Zahayu for always supporting and loving me all these years. Also, this work would not have been possible without the prayers, love, and patience of my children, Muhammad Harriz Anwar, Muhammad Hadi Asyraf and Nur Hana Azzalea. Thank you for making our life in Birmingham a memorable and joyful one.

This thesis was copy edited for conventions of language, spelling, and grammar by Express-it-write Services.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABC	Artificial Bee Colony
ACO	Ant Colony Optimisation
BA	Bees Algorithm
BAED	Bees Algorithm with Estimation Distribution
BA-HJ	Bees Algorithm with Hooke and Jeeves
CFO	Central Force Optimization
ChaosBA	Bees Algorithm with Chaos
CS	Cuckoo Search
CSS	Charged System Search
DE	Differential Evolution
EA	Evolutionary Algorithm
EDA	Estimation Distribution Algorithm
FA	Firefly Algorithm
GA	Genetic Algorithm
GA1	Genetic algorithm with co-evolution adaptation
GA2	Dominance-based selection scheme for genetic algorithm
GSA	Gravitational Search Algorithm
HJ	Hooke and Jeeves

ILS	Iterated Local Search
MBA	Mine Blast Algorithm
PSO	Particle Swarm Optimisation
PSO-DE	Particle Swarm Optimisation with Differential Evolution
qABC	Quick Artificial Bee Colony
RO	Ray Optimization
SA	Simulated Annealing
SC	Society and Civilisation algorithm
SI	Swarm Intelligence
SOA	Seeker Optimization Algorithm
SPSO2011	Standard Particle Swarm Optimisation 2011
TLBO	Teaching and Learning Based Optimization algorithm
TS	Tabu Search
UPSOm	Unified Particle Swarm Optimisation

LIST OF SYMBOLS

D	Number of dimensions
f	Function that represents the optimisation problem
max_i	Upper domain for variables
min_i	Lower domain for variables
nb	Number of best sites
nb-ne	Remaining best sites
ne	Number of elite sites
ngh	Initial size of neighbourhood
nrb	Recruited bees for remaining best sites
nre	Recruited bees for elite sites
ns	Number of scout bees
ns-nb	The remaining scout bees
stlim	Limit of stagnation cycles for site abandonment
t	the t^{th} iteration of the algorithm
x	Parameter to be optimized /design or decision variable(s), can be continuous, discrete or a mixture of both
$x^{(k)}$	Base point
$x_p^{(k+1)}$	Temporary based point for a new exploratory move
μ	Control parameter

CHAPTER 1

INTRODUCTION

1.1 Research background

To assist organisations with improving their profitability, researchers in industrial engineering have been searching for the most effective optimisation approach that maximises revenue and simultaneously minimises costs under given constraints. Over the years, numerous optimisation techniques have been developed to help engineers with tasks such as line balancing, job scheduling, process and production planning, and facility layout optimisation.

Due to the complexity of real industrial engineering problems, traditional optimisation could no longer offer the best solutions to users. The need for advanced approaches has seen the rise of metaheuristic optimisation algorithms, including the Evolutionary Algorithm (EA) and the Genetic Algorithm (GA), that can provide a reliable approximate solution. In recent years, researchers have developed a new type of metaheuristic algorithm inspired by the behaviour of biological living creatures such as ants, bees, birds, and fireflies. Among the most common algorithms of this group are Ant Colony Optimisation (ACO), Bees Algorithm (BA), Artificial Bee Colony (ABC), Particle Swarm Optimisation (PSO), and Firefly Algorithm (FA). ACO is inspired by the foraging behaviour of ants, while PSO is inspired by the behaviour of bird flocking. FA mimics the flashing behaviour of fireflies, while BA and ABC imitate the foraging behaviour of honeybees to locate sources of nectar.

1.2 Motivation

Since its establishment in 2005 (Pham et al., 2005), the Bees Algorithm has become prominent among bee-inspired algorithms. The Bees Algorithm's unique approach of combining exploitation and exploration in the search procedure of the bees in a colony produces an excellent performance comparable to that of other optimisation algorithms. Previously, the Bees Algorithm has undergone numerous enhancements that mostly focused on neighbourhood search, parameter tuning, population size, and hybridisation with other algorithms.

Although several improvements have been introduced to the Bees Algorithm, there is still a need for further developments. For instance, the current standard Bees Algorithm abandons the 'elite bees' after a few iterations according to the chosen parameter. These elite bees could possibly reach the global optimum, but the current procedure prevents the algorithm from reaching it. A direct search method used to gather information from these elite bees could intensify the local search of the algorithm to reach the optimum point.

Introducing different methods in the deployment of bees is worth exploring. Random methods currently practised might be unsuitable for the Bees Algorithm. Another important area to consider is the ability of the algorithm to use its current information to produce new promising solutions. For example, a statistical method of analysis could be used in this approach to make the Bees Algorithm converge faster.

1.3 Aim and objectives

The overall aim of this research is to advance the ability of the Bees Algorithm to find solutions for single objective optimisation problems.

The following objectives were set to attain this aim:

i. To develop an improved Bees Algorithm by incorporating Hooke and Jeeves method in the local search procedure to intensify and speed up the search.
ii. To introduce chaos in the deployment method of the bees in local and global search phases of the Bees Algorithm.
iii. To develop a version of Bees Algorithm that uses the current information of the best-so-far solutions to produce new points using estimation distribution approach.

1.4 Research methods

The methods adopted are as follows:

i. Review recent developments in swarm-based optimisation by focusing on honeybees related algorithms and learning mechanisms of other algorithms to identify current trends, research gaps, and directions for further investigation.
ii. Develop the proposed algorithms in R, an open source programming language.
iii. Evaluate the performance of the developed algorithms on a set of unconstrained continuous benchmark functions with various landscapes including unimodal and multimodal functions. The improved Bees Algorithms were further tested on constrained mechanical design problems and the results were compared with other algorithms from the literature.

iv. Analyse the comparison results using the Mann-Whitney test to examine the significance of the performance improvements achieved.

1.5 Outline of the thesis

The remainder of the thesis is organised as follows:

Chapter 2 begins by defining optimisation and swarm intelligence. The chapter reviews swarm intelligence, focusing on swarm-based algorithms particularly those mimicking the foraging behaviour of honeybees. The Bees Algorithm's concept, procedures, applications, and evolutions are also reviewed in detail.

Chapter 3 describes the incorporation of Hooke and Jeeves' method in the neighbourhood search of the Bees Algorithm. The modified Bees Algorithm is tested on a set of continuous benchmark functions and its performance is compared with the standard Bees Algorithm and the other two swarm-based algorithms. Furthermore, the modified Bees Algorithm is tested on several unconstrained benchmark mechanical design problems. The results are also compared with the standard Bees Algorithm and other algorithms from the literature.

Chapter 4 introduces the Bees Algorithm with chaos replacing the random deployment of the bees in local and global search phases. The proposed algorithm is tested on the same set of benchmark functions as in Chapter 3. Similar comparisons as in the previous chapter are also used in both unconstrained and constrained benchmark problems.

Chapter 5 presents the use of estimation distribution in the improved Bees Algorithm. This statistical method is applied to the best-so-far population to generate new promising solutions before entering the local search phase of the algorithm. Similarly, the performance of the improved version is evaluated and compared via the methods adopted in Chapter 3 and Chapter 4.

Chapter 6 summarises the contributions and conclusions drawn from this research. Suggestions for future work are presented in this chapter.

CHAPTER 2

LITERATURE REVIEW

2.1 Preliminaries

This chapter reviews the intelligent optimisation concept, primarily that based on Swarm Intelligence (SI). The review presents the learning mechanism of the canonical Bees Algorithm, past improvements to it, and its applications in real-world optimisation problems.

2.2 Global Optimisation

In almost every field of science, engineering, economics, and business, the quest to find acceptable solutions for global optimisation problem has always motivated researchers and practitioners to produce new theories and methods. From designing an expensive sports car to producing a mass production of recyclable plastic bottles, engineers need to optimise the configuration parameters that provide a balance between maximising profit, quality, safety, and efficiency; and minimising cost, defects, and energy consumption.

Global optimisation is a branch of applied mathematics and numerical analysis that focuses on optimisation by finding the optimal value of a given function among all possible solutions (local optima) in the neighbourhood of the candidate solution (Weise, 2009). Optimisation can be defined as mathematical methods in solving quantitative problems by finding the most

suitable value for an objective function within a given domain in many disciplines including engineering, physics, biology, business, and economics (Vora & Mirnalinee, 2015).

Mathematically, optimisation problems can be classified as discrete or continuous depending on the variables involved. The process of having a countably infinite set of potential solutions and searching for an optimal solution at the same time is known as a discrete optimisation process. In discrete optimisation problems, the goal is to find the best combination among the given variables. Modelling with discrete variables is part of discrete optimisation, while modelling with continuous variables is part of continuous optimisation problems. In discrete optimisation, some or all the variables in a model are required to belong to a discrete set. Conversely, continuous optimisation can take on any value within a range of values, usually real numbers. Continuous optimisation problems are typically solved using algorithms that generate a sequence value of the variables that converge to a solution of the problem. Therefore, to find a reasonably good global solution in a reasonable amount of time, various optimisation algorithms have been developed.

2.3 Optimisation Algorithms

For years, researchers have made numerous efforts to develop optimisation algorithms in solving global optimisation problems. These optimisation algorithms can be classified into deterministic and stochastic algorithms. Deterministic algorithms seek to find the global optimum with guaranteed convergence following the same computation steps. It is often used in a problem that has a clear relation between the characteristics of the possible solutions and their fitness values. However, if the relation between a solution candidate and its fitness is

unclear or too complicated, or has a high dimensionality search space, it becomes harder to solve the problem by using deterministic algorithms (Weise, 2009). Then, stochastic algorithms provide an alternative solution by offering a non-deterministic algorithm which relies on probabilistic operations (Das, Panigrahi, & Pattnaik, 2010). These algorithms use randomness in their procedures and focus only on promising areas during the search. An important class of algorithm under this stochastic type of operations is metaheuristics algorithms as shown in Figure 2.1.

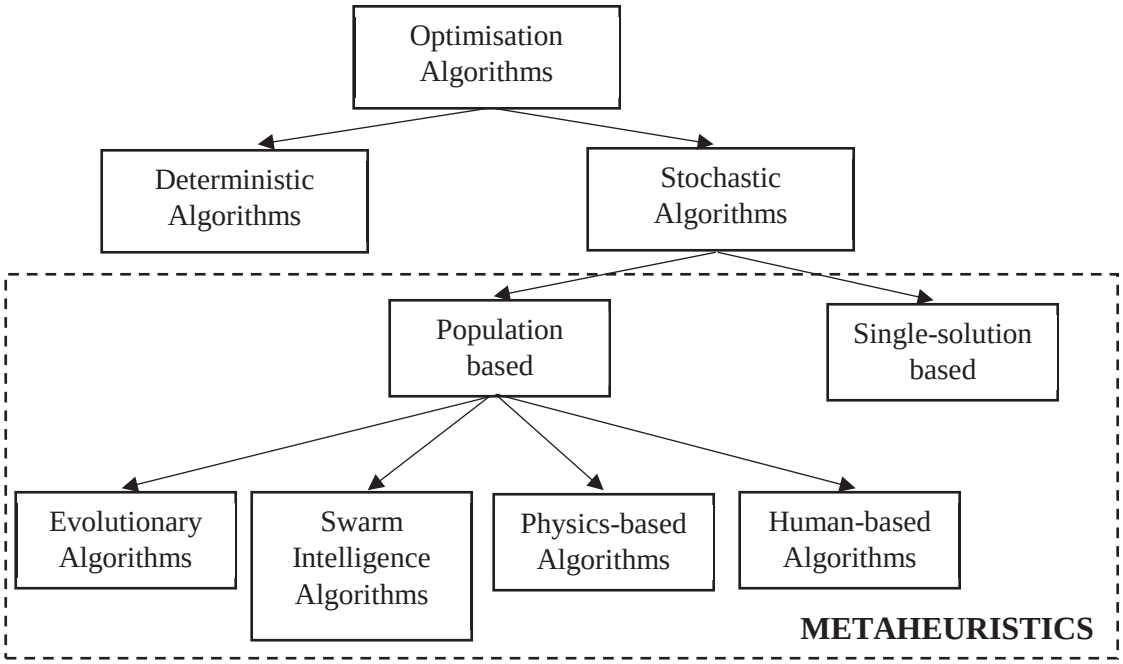


Figure 2.1: Classification of Metaheuristics in Optimisation Algorithms

The most well-known deterministic algorithms are State Space Search, Branch and Bound, and Algebraic Geometry. Stochastic algorithms can be classified into single-solution based and population-based algorithms. The most popular single-solution based algorithms are Tabu Search (TS), Simulated Annealing (SA), and Iterated Local Search (ILS). In the population-based category, the most popular evolutionary algorithms are Genetic Algorithms (GA) and Differential Evolution (DE). The most popular physics-based algorithms are Gravitational

Search Algorithm (GSA), Central Force Optimization (CFO), Charged System Search (CSS), and Ray Optimization (RO). The most popular human based algorithms include Teaching and Learning Based Optimization algorithm (TLBO), Seeker Optimization Algorithm (SOA), and Mine Blast Algorithm (MBA). Finally, the most well-known Swarm Intelligence (SI) algorithms include Ant Colony Optimisation (ACO), Particle Swarm Optimisation (PSO), Firefly Algorithm (FA), Cuckoo Search (CS), Artificial Bee Colony (ABC), and Bees Algorithm (BA).

2.4 Metaheuristics

The term "Metaheuristics" was initially introduced in 1986 by Glover in his publication to describe Tabu Search characteristics (Glover, 1986). Since then, a metaheuristics algorithm has been widely defined as a high-level algorithm that intelligently orchestrates interaction between heuristics to solve a wide range of optimisation problems (Osman & Laporte, 1996). Two major contradictory components in metaheuristics algorithms are exploration (diversification) and exploitation (intensification).

In developing a metaheuristic algorithm, these components are essential in balancing the need to maintain the diversity of the solutions while simultaneously avoiding premature convergence. Figure 2.2 (Talbi, 2009) shows the relationship between the two components and their roles in determining the type of metaheuristics algorithms. In general, the exploitation procedure guides the algorithm to thoroughly search for better solutions in a promising region to accelerate its convergence. Likewise, the exploration procedure helps the algorithm to reach unexplored regions in the hope of finding new good solutions.

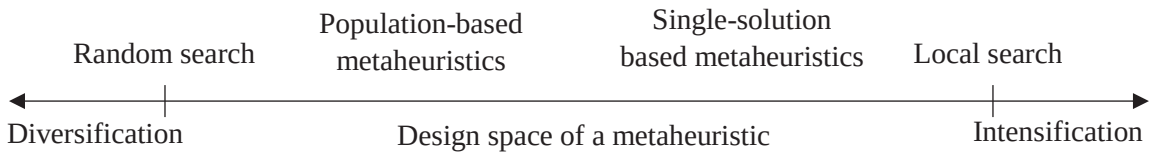


Figure 2.2: Criteria in designing a metaheuristic: diversification versus intensification

Metaheuristics algorithms can be classified based on the number of solutions being manipulated: single-solution based algorithms and population-based algorithms. Generally, single-solution or trajectory-based algorithms work on a single candidate solution and describe a trajectory in the search space during the progress of the search such as Tabu Search (TS) (Glover, 1986), Simulated Annealing (SA) (Kirkpatrick, Gelatt, & Vecchi, 1983) and Iterated Local Search (ILS) (Glover & Kochenberger, 2003). Meanwhile, population-based algorithms deal with a set of solutions and iteratively improve them through the search space. Some examples of established population-based algorithms are Genetic Algorithms (GA) (Goldberg, 1989), Differential Evolution (DE) (Storn & Price, 1997) and Swarm Intelligence (SI). Among the algorithms associated with SI include Ant Colony Optimisation (ACO) (Dorigo & Di Caro, 1999), Particle Swarm Optimisation (PSO) (Kennedy & Eberhart, 1995), Cuckoo Search (CS) (Yang & Deb, 2009), Artificial Bee Colony (ABC) (Karaboga, 2005), Firefly Algorithm (FA) (Yang, 2008), and Bees Algorithm (BA) (Pham et al., 2005).

2.5 Swarm Intelligence

In the last two decades, Swarm Intelligence (SI) algorithms, a family of population-based metaheuristics, have garnered a significant amount of interest from researchers due to their robustness and flexible behaviour in solving difficult optimisation problems. The term “Swarm

Intelligence" was popularised by Beni and Wang in their paper when discussing Cellular Robotic System (CRS), where simple agents organise themselves through neighbourhood interactions (Beni & Wang, 1993). Different from other population-based metaheuristics, SI algorithms were inspired by the collective behaviours and self-organising agents such as in colonies foraging, animal herding, bacteria growth, fish schooling, honeybees, and so on (Mavrovouniotis, Li, & Yang, 2017).

The impressive abilities of SI to organise itself without an organiser has encouraged attempts to develop algorithms inspired by swarms in nature. Hence, during the last decades, numerous algorithms have been developed and tested. Examples of algorithms inspired by SI include Particle Swarm Optimisation (PSO), Ant Colony Optimisation (ACO), Cuckoo Search (CS), Artificial Bee Colony (ABC), and Bees Algorithm (BA). Table 2.1 shows the list of SI algorithms found in the literature in the past decades.

2.5.1 Particle Swarm Optimisation (PSO)

One of the pioneer SI algorithms that received great attention in the world of optimisation is Particle Swarm Optimisation (PSO). Originated from the socially organised behaviour of bird flocks in nature, PSO is an algorithm that uses each bird as a particle representing the potential solutions around the search space according to its position and velocity. Eberhert and Kennedy proposed that the method could be used to guide the particles towards a better position using the information collected through communication with neighbouring particles (Kennedy & Eberhart, 1995).

Table 2.1: Swarm Intelligence (SI) algorithms in the literature

Entities	Swarming Behaviour	SI Algorithms
Particles	Aggregating	Particle Swarm Optimization (Kennedy & Eberhart, 1995)
Ants	Foraging	Ant Colony Optimization (ACO) (Dorigo & Di Caro, 1999)
Bees	Foraging	Marriage in Honey Bees Optimization Algorithm (Abbass, 2001), Bees Algorithm (Pham et al., 2005), Artificial Bee Colony Algorithm (Karaboga, 2005) , Bee Colony Algorithm (Teodorović et al., 2006), Bee Collecting Pollen Algorithm (Lu & Zhou, 2008)
Glowworm	Foraging	Glowworm Swarm Optimization (Krishnanand & Ghose, 2005)
Fireworks	Explosion	Fireworks Algorithm (Tan & Zhu, 2005)
Cat	Behaviour	Cat Swarm Optimization (Chu, Tsai, & Pan, 2006)
Weed	Ecological	Invasive Weed Optimization (IWO) (Mehrabian & Lucas, 2006)
Monkeys	Climbing	Monkey Search (Mucherino & Seref, 2007)
Fireflies	Gathering	Firefly Algorithm (Yang, 2008)
Cockroaches	Foraging	Roach Infestation Optimization (Havens et al., 2008)
Frogs	Jumping	Jumping Frogs Optimization (Garcia & Perez, 2008)
Masses	Gathering	Gravitational Search Algorithm (Rashedi, Nezamabadi-pour, & Saryazdi, 2009)
Cuckoos	Brooding	Cuckoo Search Algorithm (Yang & Deb, 2009)
Dolphins	Clustering	Dolphin Partner Optimization (Yang, Jiang, & Yan, 2009)
Bats	Echolocation	Bat Algorithm (Yang, 2010)
Bacteria	Growth	Bacteria Foraging Optimization (Passino, 2010)
Flies	Foraging	Fruit fly Optimization Algorithm (Pan, 2012)
Krill	Herding	Krill Herd Algorithm (Gandomi & Alavi, 2012)
Lion	Social	Lion′s Algorithm (Rajakumar, 2012)
Birds	Mating	Bird Mating Optimizer (Askarzadeh & Rezazadeh, 2013)
Wolves	Preying	Gray Wolf Optimizer (Mirjalili, Mirjalili, & Lewis, 2014)
Algae	Lifestyle	Artificial algae algorithm (AAA) (Uymaz, Tezel, & Yel, 2015)
Tree-seed	Reproduction	Tree-seed algorithm (TSA) (Kiran, 2015)
Spider	Foraging	Social Spider Algorithm (SSA) (Yu & Li, 2015)
Ions	Motion	Ion Motion Algorithm (Javidy, Hatamlou, & Mirjalili, 2015)
Whale	Foraging	Whale Optimization Algorithm (WOA) (Mirjalili & Lewis, 2016)
Dragonfly	Navigating	Dragonfly Algorithm (DA) (Mirjalili, 2016)
Moth	Pathfinding	Moth Swarm Algorithm (MSA) (Mohamed et al., 2017)
Salp	Foraging	Salp Swarm Algorithm (SSA) (Mirjalili et al., 2017)
Meerkat	Foraging & care	Meerkat Clan Algorithm (Al-obaidi, Abdullah, & Ahmed, 2018)

In the PSO algorithm, particles move around the search space according to the variation of velocity based on the individual particle's previous best position and other particle's best position. Unless the termination criteria are met, each particle is updated in each iteration by the following rule:

$$v_{ij}(t+1) = w \cdot v_{ij}(t) + c_1 \cdot r_1 \cdot \left(pbest_{ij}(t) - x_{ij}(t)\right) + c_2 \cdot r_2 \cdot \left(gbest_j(t) - x_{ij}(t)\right) \quad (1.1)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (1.2)$$

where $i(\in \{1, \dots, N\})$ is the particle's index in the swarm and $j(\in \{1, \dots, n\})$ denotes the component's index of the corresponding particle. Also, w is the positive inertia weight, c_1 and c_2 are the cognitive and social learning factors, respectively, and r_1 and r_2 are the two random numbers uniformly distributed from zero to one.

The PSO algorithm has been successfully applied to various areas of real-world applications such as telecommunications, neural network training, system simulation and identification, decision making and planning, and signal processing (Gogna & Tayal, 2013). Various attempts to improve the PSO are noted in the literature because it has simpler and fewer parameters that require tuning and is efficient in solving optimisation problems (Jordehi & Jasni, 2012). Many researchers have tried to improve the PSO by implementing new learning mechanisms inside the algorithm or combining it with other algorithms that can overcome the weaknesses of the PSO. The latest standard version of PSO is called Standard Particle Swarm Optimisation 2011 (SPSO2011). This improved version of PSO is equipped with an adaptive random topology and rotational invariance (Zambrano-Bigiarini, Clerc, & Rojas, 2013).

2.5.2 Ant Colony Optimisation (ACO)

The ACO algorithm was inspired by the collective intelligence of ants scouting for food around their nest. The development of the ACO algorithm started from the Ant System (AS) that was applied to the Traveling Salesman Problem (TSP) (Dorigo, Maniezzo, & Colorni, 1991). The core principle that underlies the mechanism of ACO is based on the pheromones deposited by the ants on their trip back to the nest after collecting food. The more pheromone is deposited means the more regular the path is used and thus, the more ants will be attracted to follow the same path to reach the highly rated food source (Dorigo & Di Caro, 1999).

In nature there is a kind of ants that can use a trail substance or pheromone to create a path from their nest to the food source, and in some way, they can optimize this path. In ACO, the algorithm uses artificial ants, and each one represents a solution to the problem in which it is applied. In order to update the positions of the artificial ants, the pheromone trails by evaporation, the levels of pheromone trails and to calculate the amount of pheromone that was deposited by the artificial ants, respectively, the ACO dynamics equations are used as follows:

$$P_{ij}^{k} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum l \in N_i^k [\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}} if l \in N_i^k \tag{1.3}$$

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}, \forall (i,j) \in L \tag{1.4}$$

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{n} \Delta\tau_{ij}^{k}, \forall (i,j) \in L \tag{1.5}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{1}{C^k}, if\ the\ arc\ (i,j)\ belongs\ to\ T^k \\ 0, otherwise \end{cases} \tag{1.6}$$

where P is the probability of an ant k to select the node j from the node i, τ is the level of pheromone trail of an arc between the nodes i and j, η is an heuristic value or the visibility from one node to another, α is an exponent that determines how much the pheromone trail influences the final probability, β is an exponent that determines the influence of the heuristic information, and N is the neighbourhood of the actual node or the nodes that the ant k has not visited yet. The rate of evaporation of the pheromone trail is given as ρ, L is a set of valid arcs of the graph, Δ denotes the amount of pheromone that the ant k deposited in the arc between the nodes i and j, based on the length of the tour of the ant k and finally, T represents the full tour constructed by the ant k, while C is the length of that particular tour.

Later on, ACO was extended and adapted for continuous optimisation by Bilchev and Parmee in their work (Bilchev & Parmee, 1995). The development of ACO for continuous optimisation was followed by a few more researchers by using probability density functions instead of discrete probability distributions in the algorithm (Socha, 2004; Socha & Dorigo, 2008; Tsutsui, 2004). Recently, new variants of ACO were introduced with improved concepts of pheromones to speed up the execution (Ismkhan, 2017) and fuzzification of the parameters in order to advance the performance (Liao & Su, 2017; Olivas et al., 2017). The ACO algorithm has also been successfully applied to several optimisation problems such as maintenance optimisation problem (Zhou et al., 2013), photovoltaic systems (Jiang, Maskell, & Patra, 2013), delimiting urban growth boundaries (Ma, Li, & Cai, 2017), and economic emission dispatch (Zhou et al., 2017).

2.5.3 Cuckoo Search (CS)

The Cuckoo Search (CS) algorithm was developed by Young and Deb to emulate the parasitic breeding behaviour of some cuckoo birds in combination with the Lévi flight behaviour of some birds and fruit flies (Yang & Deb, 2009). Brood parasitism in some cuckoo species was the main mechanism in the development of the CS algorithm. Cuckoos lay their eggs in the nest of other host birds and match them through mimicking the colour and pattern of the eggs. If the host bird discovers the eggs are not theirs, it will either throw the eggs out or simply abandon its nest and build a new one elsewhere. With the ability to hatch slightly earlier than the host eggs, the hatched cuckoo chick will evict the host eggs out of the nest and increase their share of food provided by its host bird. In the optimisation context, each egg in the nest represents a candidate solution and as the cuckoo lays only one egg, it also represents one solution. The purpose is to generate new and potentially better solutions that will replace the worse solutions in the current nest population. Furthermore, the quality of solutions is evaluated through the objective function of the problem to be solved.

The CS uses a balance between exploration and exploitation. This algorithm is equiponderant to the integration of Lévy flights. When generating new solutions x^{t+1} for a cuckoo *i*, a Lévy flight is performed according to Eq. 1.7.

$$x_i^{t+1} = x_i^t + \alpha \oplus Lévy(\lambda) \tag{1.7}$$

where, $\alpha > 0$ is the step size related to the scales of the problem of interests. The x_i^t in the above equation represents the current location, which is the only way to determine the next location x_i^{t+1}. This is called the random walk and the Markov chain. The product $\oplus$ denotes the entry

wise multiplications. A global explorative random walk by using Lévy flights can be expressed as follows:

$$Lévy{\sim}u = t^{-\lambda}, 1 \ll \lambda \ll 3 \quad (1.8)$$

Since the development of CS in 2009, much effort has been dedicated by researchers to improve the performance of the algorithm. Some recent modifications include:

i. a Modified Cuckoo Search-based Rough Sets (MCSRS) that used rough sets theory to build the fitness function based on two factors: the number of features and classification quality (Aziz & Hassanien, 2016);
ii. a new Modified Cuckoo Search Algorithm (MCSA) that proposed crossover operation to balance exploration and exploitation processes (Giridhar et al., 2017);
iii. the Adaptive Cuckoo Search Algorithm (ACSA) that focused on generating a new solution of the CS algorithm and keep the solution generation based on alien egg discovery (Dinh, Nguyen, & Vo, 2016);
iv. and a new modified version of the CS algorithm that proposed a two-stage initialisation process: divide the population into subpopulations; and combine all the best strings to a new form of the subpopulation (Rao & Venkaiah, 2017).

Various applications of the CS algorithm to real-world problems have been reported in the literature. A summary of recent applications of the CS algorithm is presented in Table 2.2.

Table 2.2: Summary of recent applications of CS algorithm in the literature

Application	Publication
Engineering design and applications	Kaveh (2017); Qu and He (2016); Mohamad et al.(2015); Ahmed and Salam (2014); Esfandiari (2014)
Power and energy	Zamani, Tavakoli, and Etedali (2017); Majumder (2016); Khoshgoftar Manesh and Ameryan (2016); Sanajaoba and Fernandez (2016); T. T. Nguyen, Vo, and Dinh (2016); Nguyen and Vo (2016); Sudabattula and Kowsalya (2016); Abd Elazim and Ali (2016); Devabalaji, Yuvaraj, and Ravi (2015)
Image processing	Sri Madhava Raja and Vishnupriya (2016); Bhandari et al.(2014)
Economic dispatch problems	Pham et al.(2016); Sekhar and Mohanty (2016)
Clustering and mining	Amsaleka and Latha (2014); Cobos et al. (2014); Abbas et al.(2014)
Medical	Chatterjee et al.(2017); Liu and Fu (2014)

2.5.4 Artificial Bee Colony (ABC)

ABC was introduced by Karaboga and was inspired by the foraging behaviour of honeybees in nature based on the concept proposed by Tereshko and Loengarov in their study on collective decision making of honeybees (Tereshko & Loengarov, 2005). In the ABC algorithm, the location of food source represents the candidate solution to the problem being considered while the amount of nectar found in the food source denotes the quality of the solution. The colony of artificial bees in the ABC algorithm contains three kinds of bees: employed bees, onlooker bees, and scout bees. Employed bees is a group of bees that is associated with food sources (candidate solutions), nectar amounts (quality of the candidate solutions) and sharing the information (through the waggle dance) with onlooker bees in the hive. Onlooker bees is the group of bees waiting in the hive and deciding the location of the food source by watching the employed bees dancing. Meanwhile, scout bees are the ones searching for food randomly across the search space (Karaboga, 2005; Karaboga et al., 2014).

The general algorithmic structure of the ABC approach started with the initialisation phase according to Eq. (1.9),

$$x_{m,i} = l_i + rand(0,1) * (u_i - l_i) \tag{1.9}$$

where *i* is the dimension of the food source m, while l_i and u_i are the lower and upper bounds of the parameter $x_{m,i}$, respectively. Employed bees search for new food sources (v_m) having more nectar within the neighbourhood of the food source x_m in their memory. They find a neighbour food source and evaluate its fitness. The neighbour food source is determined using,

$$v_{m,i} = x_{m,i} + \phi_{m,i}(x_{m,i} - x_{k,i}) \tag{1.10}$$

where x_k is a randomly selected food source, *i* is a randomly chosen parameter index and $\phi_{m,i}$ is a random number within the range of $[-1,1]$. After producing the new food source v_m, its profitability is calculated, and a greedy selection is applied between v_m and x_m . The fitness of the solution *fit* (x_m) can be calculated using,

$$fit(x_m) = \begin{cases} \dfrac{1}{1 + f(x_m)} \; if \; f(x_m) \geq 0 \\ 1 + abs\big(f(x_m)\big) \; if \; f(x_m) \geq 0 \end{cases} \tag{1.11}$$

where $f(x_m)$ is the objective function value of solution x_m. Employed bees share their food source information with onlooker bees waiting in the hive. Depending on this information, the onlooker bees probabilistically choose their food sources. The probability value p_m with which x_m is chosen by an onlooker bee can be calculated using,

$$p_m = \frac{fit(x_m)}{\sum_{m=1}^{SN} fit(x_m)} \tag{1.12}$$

After a food source for an onlooker bee is probabilistically selected, a neighbour source v_m is determined by using Eq. (1.10), and its fitness value is computed.

Since the invention of the ABC algorithm, numerous applications and extensions of the algorithm have been developed. Applications range from different areas in neural networks, engineering, image processing, data mining, sensor networks, and protein structure (Karaboga & Akay, 2009). As in other optimisation algorithms, some modifications to the ABC are still necessary to significantly improve its performance. The scientific community has proposed various improvements to the original ABC including a novel algorithm called Quick Artificial Bee Colony (qABC) that has a new position update equation for onlooker bees. The qABC also introduced a new method in site abandonment and a new parameter for neighbourhood radius (Karaboga & Gorkemli, 2012). Some of the recently improved versions of ABC are:

i. a self-adaptive ABC algorithm based on the global best candidate (SABC-GB) that was initialised by adopting chaotic systems and opposition-based learning method, by modifying the employed bee phase, and by using a probabilistic method in the onlooker bee phase (Xue et al., 2017);
ii. a new variant named ABC with memory algorithm (ABCM) which has a memory mechanism to let the artificial bees memorise their previous successful experiences (Li & Yang, 2016); and
iii. an ABC with multiple solution update rules for each employed bees or onlooker bees to obtain candidate solutions (Kiran et al., 2015).

2.6 The Canonical Bees Algorithm

Inspired by the food foraging behaviour of honeybees in nature, a group of researchers from Cardiff University successfully developed the Bees Algorithm in 2005. The algorithm performed a combination of exploitation and exploration to search for the best solution to a

given optimisation problem (Pham et al., 2005). In general, the bees represent problem variables in vector form and the food sources visited by the bees symbolize the candidate solutions to the problem.

Extensive research on honeybee behaviour has been conducted for years by various researchers around the world. Areas of interest cover hive selection, mating behaviour, and food foraging process in a bee colony. In the honeybee food foraging activity, the process starts with part of the population despatched to search for food sources near the hive. This group of bees are called scout bees and they return to the hive with nectar and information of the visited flower patches. Apart from the quality of the nectar collected, the bees transmitted vital information such as the location and distance from the hive. This information is shared with other bees in the population through a movement called “waggle dance” inside the hive. Then, the bees from the population is recruited by the scout bees to form a group of bees to be directed to the visited flower patch with high quality nectar. More bees are recruited to flower patches with higher quality nectar. The recruited bees will return to the hive and the process of sharing information and recruitment will continue.

In the canonical Bees Algorithm, several control parameters were required for the user to start the search. Table 2.3 shows the control parameters of the canonical Bees Algorithm (Pham & Castellani, 2015). The algorithm starts with scout bees (*ns*) randomly scattered across the search space. The number of scout bees represents the initial population for the search. Then, each site visited by the scout bees is evaluated by the fitness function to measure the quality of the candidate solutions. Next, the scout bees are ranked according to the quality of the fitness values. The higher the fitness value the higher the rank of the candidate solutions. Further, the

best site (*nb*) is selected from the ranked population. This is the beginning of the exploitation phase of the algorithm. The top-rated sites in the selected best sites (*nb*) are chosen as elite sites (*ne*) and the remaining best sites are noted as (*nb-ne*). Subsequently, the elite sites (*ne*) and remaining best sites (*nb-ne*) recruit other bees from the hive to exploit their locations according to *nre* and *nrb* respectively. The value of *nre* is always greater than *nrb* in this case (*nre* > *nrb*). At this stage, the recruited bees (*nre* and *nrb*) are randomly placed across the neighbourhood of the *ne* and *nb-ne*, respectively. Then, the fitness values of the *nre* and *nrb* are evaluated and the best recruited bees for each neighbourhood are selected as the new scout bees in the population. In this process, only the fittest scout bee is retained from each neighbourhood of the *nb.* The newly selected scout bees (from the neighbourhood of *ne* and *nb-ne*) are returned to the hive to share the current fittest solution.

Table 2.3: Bees Algorithm parameters

Parameter	Description
ns	Number of scout bees
ne	Number of elite sites
nb	Number of best sites
nre	Recruited bees for elite sites
nrb	Recruited bees for remaining best sites
ngh	Initial size of neighbourhood
stlim	Limit of stagnation cycles for site abandonment

In order to increase the search accuracy and to avoid unnecessary computations in the local search phase previously discussed, two strategies called neighbourhood shrinking and site abandonment were introduced (Pham & Castellani, 2009). In neighbourhood shrinking strategy, the patch size (*ngh*) is initially set to a large value and as the search continues, the *ngh* shrinks if the recruited bees fail to produce better fitness. However, the value of *ngh* is kept constant if the recruited bees provide better fitness values than the scout bees in the neighbourhood. The formula for the shrinking procedure is set as follows:

$$a_i(t) = ngh(t) * (max_i - min_i) \quad (13.13)$$

$$ngh(t+1) = 0.8 * ngh(t) \quad (1.14)$$

where *t* denotes the t^{th} iteration of the algorithm. The second strategy, the site abandonment procedure, is applied when the neighbourhood strategy fails to produce fitness improvement after a predefined number of consecutive stagnation cycles (*stlim*). At this stage, the local search is assumed to be at the local fitness peak and the search is terminated; thus, a new random solution is generated. Furthermore, if the abandoned site has the best-so-far fitness value, the corresponding site would be chosen as the final solution.

In the next stage, the algorithm performs an exploration phase as the global search is executed to search for candidate solutions. The remaining scout bees, *ns-nb* are randomly placed across the search space to search for new promising sites. Finally, the new population is formed by combining the best bees (*nb*) from the local search and the newly found scout bees from the global search (*ns-nb*). The stopping criterion for the algorithm could be either the completion of a predefined number of iterations or when the solution fitness is above a predefined threshold. The flow chart of the canonical Bees Algorithm is shown in Figure 2.3.

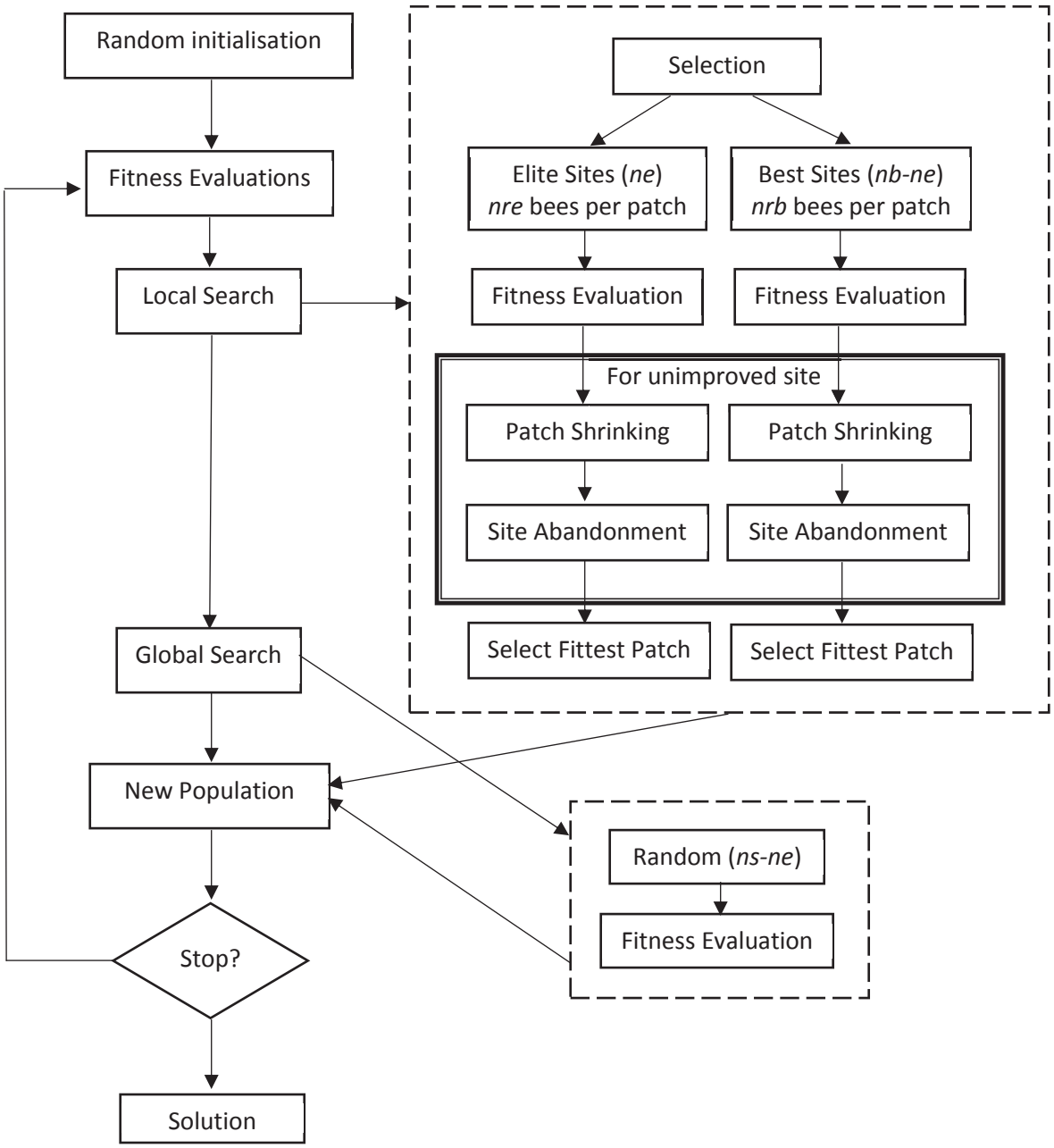


Figure 2.3: Flow chart of standard Bees Algorithm

2.7 Improvements and Applications

Since the introduction of the algorithm in 2005, many improvements have been conducted worldwide to advance the performance of the Bees Algorithm. This section will review the advancements of the algorithm and the applications carried out in the literature. Figure 2.4 shows the number of improved Bees Algorithm variants developed by researchers compared to the total publications based on the Bees Algorithm from 2005 to 2017.

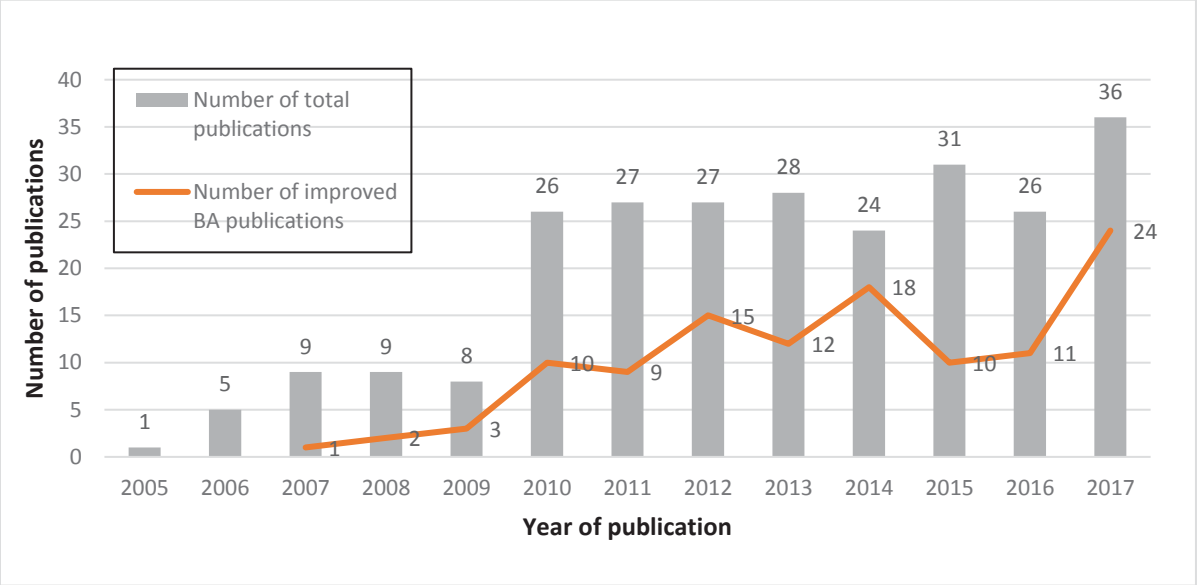


Figure 2.4: Number of total publications and improved BA found in literature

In the early years of the algorithm (2005-2006), the publications focused more on testing out the algorithm performance in the real-world applications. Pham et al. (2006) applied the Bees Algorithm in Multi-Layered Perceptron (MLP) training for control chart pattern recognition. The application of Bees Algorithm in control chart pattern recognition was extended by Pham et al. (2006a) and Pham et al. (2006b) to train Learning Vector Quantisation Networks and Radial Basis Function Networks. In addition, Pham et al. (2006) applied the usage of Bees Algorithm in the wood defect detection optimisation problem.

From 2007 to 2009, six publications had been published on the improvements of the algorithm. The total publications have also increased to around eight publications each year. The first improvement had been proposed by Pham, Castellani, and Ghanbarzadeh (2007) for the application of computer-aided preliminary design. The proposed algorithm was intended to increase algorithm performance and avoid unnecessary computations by introducing two procedures called “shrinking method” and “abandon sites without new information”. Furthermore, Pham, Castellani, and Fahmy (2008), and Pham and Castellani (2009) applied this enhanced version in the robot manipulator problem and numerical optimisation problems, respectively. Meanwhile, Pham and Darwish (2008) introduced an enhanced version of the Bees Algorithm with fuzzy greedy selection in its local search procedure. This version of Bees Algorithm automated the selection and recruitment processes, thus reducing the number of parameters. Pham and Kalyoncu (2009) conducted the application of this version of Bees Algorithm to optimise the parameters of the controller for a flexible single-link robot arm. On the other hand, Packianather et al. (2009) proposed an enhanced Bees Algorithm based on the pheromone-based communication system of honeybees in the recruitment process, thus making the algorithm more dynamic compared to the basic version. Moreover, the number of paramaters were also reduced.

The applications of the Bees Algorithm were also increased during the same period of time. Publications on applications were extended to multiobjective problems such as Printed Circuit Board (PCB) assembly (Pham, Otri, & Darwish, 2007), mechanical design (Pham & Ghanbarzadeh, 2007), Environmental/Economic Dispatch (EED) (Lee & Darwish, 2008; Pham, Lee, Darwish, & Soroka, 2008), Multiple-Input Multiple-Output (MIMO) communication (Sayadi et al., 2009) and Flexible AC Transmission System (FACTS) devices allocation (Mohamad Idris, Khairuddin, & Mustafa, 2009). The Bees Algorithm has also been

utilised in linear antenna arrays (Guney & Onay, 2007; Guney & Onay, 2008), data clustering (Pham, AL-Jabbouli, Mahmuddin, Otri, & Darwish, 2008; Pham, Otri, Afify, Mahmuddin, & Al-Jabbouli, 2007), scheduling (Lara, Flores, & Calderon, 2008; Pham, Koç, Lee, & Phrueksanant, 2007), fuzzy logic controller (Pham, Darwish, Eldukhri, & Otri, 2007), wood defect classification (Pham et al., 2007) and protein confirmational search (Bahamish, Abdullah, & Salam, 2008). Another application of the Bees Algorithm was in manufacturing cell formation (D. T. Pham, Afify, & Koç, 2007), Proportional-Integral-Derivative (PID) controller tuning (Jones & Bouffet, 2008), motion planning for robot arm (Ang, Pham, & Ng, 2009) and mechanical design optimisation (Pham et al., 2009).

Since 2010, publications on the Bees Algorithm have escalated to at least 24 publications per year and the publications on Bees Algorithm improvement have shown a significant increase. 2017 had shown the most publications of improvement with 24 improvements followed by 2014 with 18 improvements and 2012 with 15 improvements. The summary of the literature review on the major improvements from 2010 to 2017 is illustrated in Table 2.4. In recent years, the applications of BA were also expanded to environmental contamination and toxilogy (Farajvand et al., 2018), software testing (Zabil, Zamli, & Lim, 2018), wheeled mobile robot path planning (Haj Darwish, Joukhadar, & Kashkash, 2018), thermoelectric materials (Uysal et al., 2017), remanufacturing (Zheng et al., 2017), biopartitioning micellar chromatography (Zarei, Atabati, & Ahmadi, 2017), grid independent hybrid renewable energy systems (Maleki, 2017), contaminant hydrology (Mehdinejadiani, 2017), power generation of wind turbine (Assareh & Biglari, 2016), and retailing industry (Martino et al., 2016).

Table 2.4: Summary of the improvements on Bees Algorithm

Author	Improvement	Type of improvement	Type of application
Mahmudin and Yusof (2010)	Proposed a hybrid called K-Bees algorithm to find near-optimal centroids.	Hybridisation: K-means algorithm	Numerical benchmarking
Özbakir et al. (2010)	Proposed initialisation with GRASP algorithm and introduced re-initialisation mechanism for the scout bees.	Learning mechanism: Initialisation & local search	Generalized assignment problems (GAP)
Li et al. (2010)	Proposed initialisation using controlled randomisation and frequency memory method based on polar coordinate. Using the neighbourhood search by dividing the circle into 3 groups by the radius.	Learning mechanism: Initialisation & local search	Large-scale layout optimization
Ang et al. (2010)	Proposed a version of Bees Algorithm with TRIZ-inspired operators to reduce the amount of time required for PCB assembly.	Learning mechanism: Local search & global search	PCB Assembly
Pham and Darwish (2010)	Proposed a combined enhanced fuzzy greedy selection system and Kalman filtering to update the positions of recruited bees in local search phase of Bees Algorithm	Learning mechanism: Local search	Wood defects identification
Taroq Sadiq and Ghazi Hamad (2010)	Proposed a hybrid version called Bees Simulated Annealing (BSA) which used Simulated Annealing in the selection method of the recruited bees to form next population.	Hybridisation: Simulated Annealing algorithm	Four colour map problem and classical transportation problem.
Xu et al. (2011)	Proposed a version of Bees Algorithm called Adaptive Bees Algorithm (ABA). The ABA used an adaptive patch adjustment method according to the source and the rate of change of the current optimum	Learning mechanism: Local search	Semi-track aircushion vehicle (STACV) fuel economy optimisation

Table 2.4: Summary of the improvements on Bees Algorithm (continued)

Author	Improvement	Type of improvement	Type of application
Muhamad et al. (2011)	Proposed an enhanced Bees Algorithm with Local Search Manoeuvres (LSM) recruitment factor that increase or decrease the neighbourhood size in different dimensions.	Learning mechanism: Local search	Numerical benchmarking
Dereli and Das (2011)	Proposed a hybrid Bees Algorithm called hybrid-BA. A heuristic filling procedure based on the 'wall building' approach is utilised in the hybridisation.	Hybridisation: Filling procedure	Container Loading (CL) optimization problem
Q. T. Pham, Pham, and Castellani (2012)	Introduced a modified Bees Algorithm with the addition of young bees concept that are protected from competition with the selected bees until they become adults. A statistical based procedure for parameter tuning was also introduced for the modified algorithm.	Learning mechanism: Local search & global search, parameter tuning	Numerical benchmarking
Alfi and Khosravi (2012)	Proposed a hybrid algorithm called BA-SD combining Bees Algorithm (BA) and Steepest Descent (SD). The SD used the global best found in BA to search for new solution.	Hybridisation: Steepest Descent (SD)	Numerical benchmarking
Sadiq and Hamad (2012)	Proposed a new Exploration Balanced Bees Simulated Annealing Algorithm (EBBSAA) to solve combinatorial optimisation problems.	Hybridisation: BSA & EBBA	Numerical benchmarking
Abdullah and Alzaqebah (2013)	Proposed a modified Bees algorithm with disruptive selection strategy, adaptive neighbourhood structure and Basic Late Acceptance Hill Climbing (Basic LAHC) hybridisation.	Learning mechanism: Local search & global search Hybridisation: Basic LAHC	Examination timetabling problems (ETTPs)

Table 2.4: Summary of the improvements on Bees Algorithm (continued)

Author	Improvement	Type of improvement	Type of application
Shatnawi, Sahran, and Mohammad Faidzul (2013)	Proposed local and global memories in Bees Algorithm to mimic the natural behaviour of honeybees.	Learning mechanism: Local search & global search	Numerical & engineering design benchmarking
Yuce et al. (2013)	Introduced an adaptive neighbourhood size change and site abandonment strategy in the local search.	Learning mechanism: Local search	Numerical benchmarking
Hussein, Sahran, and Sheikh Abdullah (2014)	Introduced Patch-Lévy-based Initialisation Algorithm (PLIA) that models the patch concept and Lévy motion for the initialisation phase in Bees Algorithm.	Learning mechanism: Initialisation	Numerical benchmarking
Tsai (2014)	Proposed a Novel Bees Algorithm (NBA) that uses stochastic self-adaptive neighbourhood search.	Learning mechanism: Local search	Numerical benchmarking
Packianather et al. (2014)	Proposed a novel Genetic Bees Algorithm (GBA) that has genetic operators called Reinforced Global Search and a Jumping Function.	Hybridisation: Genetic Algorithm	Single Machine Scheduling problem
Lien and Cheng (2014)	Proposed a hybrid algorithm called Particle Bee Algorithm (PBA) that integrates the respective advantages of Particle Swarm Optimisation (PSO) and Bees Algorithm (BA).	Hybridisation: Particle Swarm Optimisation	Tower Crane Layout (TCL) optimisation problem
Jana, Sil, and Das (2015)	Proposed an Adaptive Polynomial Mutation based Bees Algorithm (APM-BA) that mutates each of best scout bees with adaptive polynomial mutation technique.	Learning mechanism: Local search	Protein Structure Prediction (PSP) problem

Table 2.4: Summary of the improvements on Bees Algorithm (continued)

Author	Improvement	Type of improvement	Type of application
Hussein et al. (2015)	Proposed a new version called Patch-Lévy-based Bees Algorithm (PLBA) that incorporates PLIA in initialisation phase, Greedy Lévy-based Local Search Algorithm (GLLSA) in local search and new Levy-based method for global search.	Learning mechanism: Initialisation, Local search & Global Search	Protein Structure Prediction (PSP) problem
Xie et al. (2015)	Proposed a new variation of Bees Algorithm call Forager Adjustment Strategy-Bees Algorithm (FAS-BA) that adaptively manages the forager allocation in the algorithm.	Learning mechanism: Parameter tuning	Numerical benchmarking & Resource Service Composition (RSC) problem
Yuce et al. (2015)	Proposed an improved version of Bees Algorithm (BA) based on Slope Angle Computation and Hill Climbing Algorithm (SACHCA) strategy.	Learning mechanism: Local search	Numerical benchmarking & Single Machine Scheduling
Nguyen (2015)	Proposed a novel algorithm called Hybrid SFL-Bees Algorithm that combined the strenghts of SFLA and BA to find global optimum.	Hybridisation: Shuffled Frog Leaping Algorithm (SFLA)	Numerical benchmarking
Zhou et al. (2016)	Proposed an improved Bees Algorithm with dynamic colony size, Balance Search Technique (BST) based local search, Hill Valley (HV) method based global search and two new procedures of radius estimation and optima elitism.	Learning mechanism: Initialisation, Local search & Global Search	Numerical benchmarking
Martino et al. (2016)	Proposed a new hybrid algorithm called Tabu-Bees Algorithm that utilised Tabu Search in the selection of the elite and the best patches to avoid going back to previously visited points.	Learning mechanism: Local search	Fashion retail replenishment problem

Table 2.4: Summary of the improvements on Bees Algorithm (continued)

Author	Improvement	Type of improvement	Type of application
Xu et al. (2016)	Proposed a multi-objective optimisation algorithm called Enhanced Pareto Bees Algorithm (EPBA) that adopts adaptive variable neighborhood search, local depth search based on the critical path, crossover and mutation operators, fastnon-dominated sorting, and local depth search strategies.	Learning mechanism: Local search	Manufacturing equipment services scheduling problem
Al-Araji (2016)	Proposed a novel hybrid algorithm called Bees-Slice Genetic Algorithm (BSGA). The BSGA combined the effectiveness of Slice Genetic (SG) algorithm in the local search procedure	Learning mechanism: Local search	Inverted pendulum system problem
Yuce et al. (2017)	Introduced a hybrid named Genetic Bees Algorithm (GBA) that includes two strategies into the Bees Algorithm. The strategies are "reinforced global search" and "jumping function"	Learning mechanism: Global search	Single Machine Scheduling problem
Nasrinpour, Bavani, and Teshnehlab (2017)	Proposed a new version of Bees Algorithm with fewer parameter setting called Grouped Bees Algorithm (GBA). In the GBA, the bees are grouped to search different patches with various neighbourhood sizes.	Learning mechanism: Local search & Global Search	Numerical benchmarking
Al-Araji and Yousif (2017)	Proposed a new hybrid called Hybrid Bees-PSO (HBPSO) algorithm. This combination uses PSO and Bees Algorithm ability in the local search and global search phase respectively	Hybridisation: Particle Swarm Optimisation (PSO)	Dynamic wheeled mobile robot
Ghiasi et al. (2017)	Proposed a new variant of Bees Algorithm with new method called dynamic recruitment, proportional shrinking for selected sites and site abandonment.	Learning mechanism: Local search & Global Search	Load forecasting

Table 2.4: Summary of the improvements on Bees Algorithm (continued)

Author	Improvement	Type of improvement	Type of application
Tandis and Assareh (2017)	Introduced a new hybrid algorithm called Genetic-based Bees Algorithm (GBBA). This hybrid uses crossover and neighborhood searching operators, from GA and BA, respectively, to create an algorithm with good performance in accuracy and speed convergence.	Hybridisation: Genetic Algorithm	Inverse Shape Design (ISD) problem

Based on the summary in Table 2.4, the improvements of the BA are focusing more on the learning mechanism of the algorithm. The majority areas of improvements include the initialisation, local search and global search phase in the BA. Some of the newly improved BA incorporates various concepts either from other algorithms or fundamental theories in optimisations. The hybrid BA algorithms have been proven successful in solving a wide range of applications. These include a myriad of different areas, such as numerical benchmarking, scheduling, forecasting, protein prediction, facilities planning, product design and many others.

Another type of improved version of BA is through the hybridisation techniques. Concentrating on strengths and weaknesses, exploitation and exploration criteria of the algorithm, researchers have combined the merits of few algorithms into the framework of BA to produce a profitable synergy. Amongst them are the GA and PSO algorithms. Hybridising other algorithm in the mechanism of the BA produced different characteristics of algorithm with improved performances and extended capabilities. However, there are few issues concerning the hybrid algorithms. For example, most of them will increase the number of parameters in the

algorithms, thus making it harder to tune their parameters. Furthermore, due to the complexity of hybrids, they are slightly harder to be implemented, and thus more prone to error.

2.8 Summary

This chapter was aimed at giving a glimpse of optimisation algorithms and focusing on population-based algorithms, predominantly those in the Swarm Intelligence category. The chapter also reviewed the Bees Algorithm concept, search method, and procedures. Furthermore, the chapter discussed the development and applications of the Bees Algorithm in various fields of real optimisation problems.

CHAPTER 3

THE BEES ALGORITHM WITH HOOKE AND JEEVES METHOD (BA-HJ)

3.1 Preliminaries

Honeybees are one of the many species of animals that benefit from behavioural specialisation, a significant element in SI based algorithms. Forager bees in a colony specialised in finding food search the area surrounding the hive and bring food back to the hive, recruiting other bees to help with the food collection process. More bees are recruited to go to areas that are richer in food. This unique foraging behaviour was adopted in the development of the Bees Algorithm to solve optimisation problems. The work presented here aims to propose an enhanced version of the Bees Algorithm with the Hooke and Jeeves' (HJ) method for continuous global optimisation problems. In this work, the HJ method was incorporated into the BA to intensify the existing BA local search mechanism. The performance of the proposed algorithm was tested on a set of continuous benchmark functions and mechanical design optimisation problems.

The chapter is organised as follows. Section 3.2 presents the strategy of HJ and the proposed method to be incorporated into the BA. Section 3.3 explains the experimental setup for unconstrained benchmark functions tests followed by the results and discussion in Section 3.4. In Section 3.5, the experimental setup and results for several constrained mechanical engineering benchmark problems are presented and discussed. Section 3.6 concludes the chapter.

3.2 Bees Algorithm with the Hooke and Jeeves (BA-HJ)

The ability of the BA to get to the optimum value is largely dependent on its local search and global search unique procedures as described in the previous chapter. The importance of the local search procedure to exploit promising candidate solutions is an easy target to try to improve the way the algorithm works. In the standard BA procedure, the local search starts with randomly deploying recruited bees in the neighbourhood of the best bees and finally select the bee that provides a better solution. Using the same procedures, in this proposed algorithm, a strategy called Hooke and Jeeves (HJ) is incorporated into BA to intensify the local search mechanism.

The Hooke and Jeeves (HJ) strategy was originally introduced in 1961 (Hooke and Jeeves 1961). In this strategy, they used a 'direct search' method to solve problems related to optimisation. The 'direct search' method does not depend upon the knowledge of the background of the objective function. The HJ method consists of a sequence of exploratory moves about a base point $x^{(k)}$ that, if successful, is followed by pattern moves (Lai & Chan, 2007). Exploratory moves are performed to look for an improving direction in which to move while pattern moves are a larger search in the improving direction. Larger and larger moves are made if the improvement continues. The new point for the pattern move is calculated as:

$$x_p^{(k+1)} = x^{(k)} + \left(x^{(k)} - x^{(k-1)}\right) \tag{14.1}$$

where $x_p^{(k+1)}$ is temporary base point for a new exploratory move.

The HJ strategy is detailed as follows (Moser & Chiong, 2009):

Step 1: Obtain an initial base point $x^{(k)}$. Determine the set of step lengths.

Step 2: Move the base point along each of the D-dimensional axes at a time and evaluate the result. Adopt a new point if there is improvement on the previous point. This takes at least D, at most 2D evaluations. If any of the moves are successful, go to Step 3. If none is successful, go to Step 4.

Step 3: Repeat the successful moves in a combined pattern move. If the new point has better fitness, assume it as the new base point. Return to Step 2, whichever the outcome.

Step 4: Adjust step length to the next smaller step. If there is a smaller step, continue from Step 2. If not, terminate.

The HJ procedure repeats until no improvement can be made in any dimension. The step size is reduced, and the procedure is repeated until there are no more step sizes.

To enhance the capability of standard BA, the proposed methodology introduces the HJ method in the local search procedure as shown in Figure 3.1. The HJ method was implemented to intensify exploitation in the most promising neighbourhood, elite sites. In this approach, the HJ method uses the location of the best-so-far solution as the starting point. The new location obtained by the HJ method is recorded as the new elite sites for the current population. The new elite sites will undergo the same procedure as in the standard BA.

The main steps of the proposed algorithm known as Bees Algorithm with Hooke and Jeeves (BA-HJ) are summarised as follows. After the elite sites identify the best so far solution using fitness evaluation and the ranking procedure, the HJ procedure is executed. A local search is activated using the current best solution as the base point until the predetermined maximum function evaluations is reached. Next, the new best solution is selected to form the new population. Finally, the new population is evaluated as in the standard BA to complete the cycle.

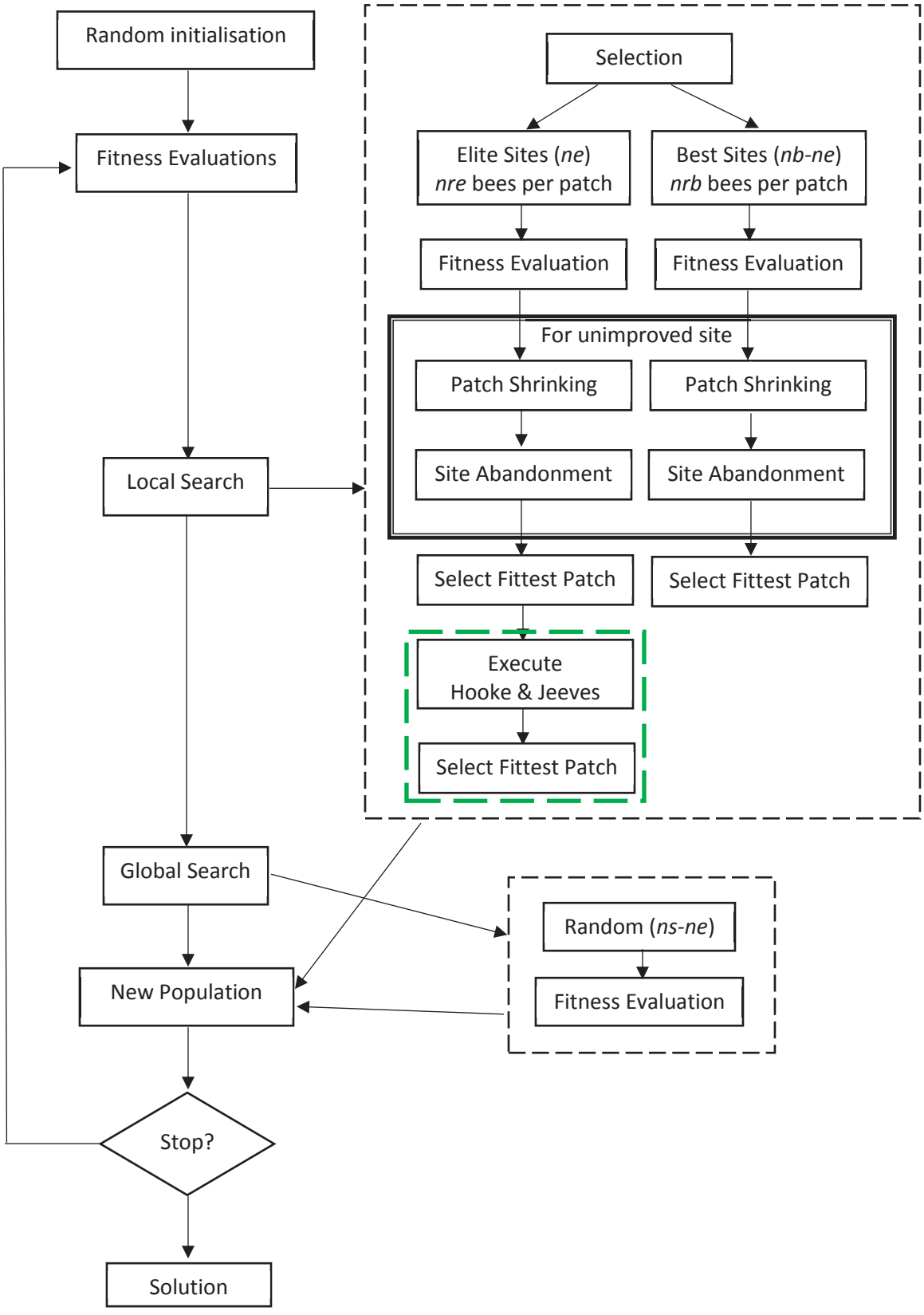


Figure 3.1: Flow chart of Bees Algorithm with Hooke and Jeeves (BA-HJ)

3.3 Experimental Setup

In this section, the proposed algorithm was applied on a set of 15 continuous benchmark functions collected from several references to evaluate its performance. The benchmark functions are listed in Table 3.1 and Appendix A. This set includes unimodal functions (f_1, f_2, f_7-f_{10}) and multimodal functions (f_3-f_6, f_{11}-f_{15}), more details about these functions are reported in (Jamil & Yang, 2013; Karaboga & Akay, 2009; Mirjalili, Mirjalili, & Lewis, 2014). The selection of these functions was based on the variety of their characteristics and surface landscapes. All the tests were executed on a computer with an Intel Xeon 2.40 GHz processor and 8GB of RAM.

Table 3.1: Benchmark functions name and features

Function	Name	Feature
f_1	Martin & Gaddy	Unimodal
f_2	Booth	Unimodal
f_3	Goldstein & Price	Multimodal
f_4	Schaffer	Multimodal
f_5	Six Hump Camel	Multimodal
f_6	Michalewicz	Multimodal
f_7	Hypersphere	Unimodal
f_8	Rosenbrock	Unimodal
f_9	Powell	Unimodal
f_{10}	Axis	Unimodal
f_{11}	Ackley	Multimodal
f_{12}	Griewank	Multimodal
f_{13}	Rastrigin	Multimodal
f_{14}	Zakharov	Multimodal
f_{15}	Styblinski-Tang	Multimodal

The algorithms were coded using the R programming language. The parameters adopted in the implementation of the standard BA and the proposed algorithm are listed in Table 3.2. R package 'dfoptim' (Varadhan & Borchers, 2011) was used in the implementation of HJ for the proposed algorithm with a maximum number of objective function evaluations of 200. All the algorithms were run for 50 times until either the minimum of the function was approximated to be better than 0.001 or a maximum number of function evaluations (500,000) elapsed. The following subsections present the comparison results obtained on the given test benchmark functions between the proposed algorithm and standard BA.

To further validate the proposed algorithm, comparisons were also made with two well-known swarm-based algorithms, the Standard Particle Swarm Optimisation 2011 (SPSO2011) and the Quick Artificial Bee Colony (qABC). The selection of SPSO2011 was based on the criteria of being one of the most popular swarm-based algorithms available in the literature while qABC was chosen based on the similarity in the concept of the honeybee foraging behaviour used in the algorithm. The parameter settings of SPSO2011 and qABC were based on the original publications by Zambrano-Bigiarini, Clerc, and Rojas (2013) and Karaboga and Gorkemli (2012) respectively. However, for a fair comparison, the number of population size for both algorithms was set to 100 according to BA-HJ parameter setting.

Table 3.2: Parameter setting for the BA-HJ and the standard Bees Algorithm

Parameter	Value
Number of scout bees, *ns*	26
Number of elite sites, *ne*	2
Number of best sites, *nb*	6
Recruited bees for elite sites, *nre*	20
Recruited bees for remaining best sites, *nrb*	10
Initial size of neighbourhood, *ngh*	0.01
Limit of stagnation cycles for site abandonment, *stlim*	10

3.4 Results and Discussion

In this section, the performances of BA-HJ and standard BA are recorded in Table 3.3. As shown in Table 3.3, both algorithms were evaluated according to the number of success runs, median accuracy of the final solutions and median speed of the algorithms to get to the global optimum of the functions. In this approach, the closer the accuracy to zero, the more accurate the solution was, and the lesser the speed of the algorithm, the faster the algorithm attained the optimum value.

For statistical comparison purposes, the Mann-Whitney statistical significance test was used in this study. To determine whether the difference between the medians was statistically significant, the *p*-values for accuracy and speed obtained from the set of benchmark functions were compared at $\alpha = 0.05$ significance level and recorded in Table 3.4. The difference between the population medians is statistically significant if the *p*-value is less or equal to 0.05. Contrarily, there is not enough evidence to conclude the difference between the population medians if the *p*-value is greater than 0.05. The algorithm that shows significance over the other is recorded in boldface.

As shown in Table 3.3, the BA-HJ outperformed the standard BA in thirteen out of fifteen benchmark functions tested in this study. By referring to Table 3.4, only one function (f_4, *Schaffer*) showed a non-significant relationship between standard BA and BA-HJ. This indicated that the performance of both algorithms was comparable for this function. Figure 3.2 shows the convergence graphs of standard BA and BA-HJ for the benchmark functions. These

convergence graphs display the performance of the algorithms in one out of the fifty runs for each benchmark function.

Table 3.3: Performance comparison between standard Bees Algorithm and BA-HJ

Function	Standard Bees Algorithm			BA-HJ		
	Success	Accuracy	Speed	Success	Accuracy	Speed
f_1	50	3.90E-04	1376	50	1.42E-08	**311**
f_2	50	3.81E-04	1226	50	5.48E-08	**287.5**
f_3	50	5.06E-04	1826	50	7.09E-06	**275**
f_4	50	2.39E-04	7583.5	50	1.54E-11	7584.5
f_5	50	3.90E-04	926	50	2.84E-05	**271.5**
f_6	50	5.20E-04	96096.5	50	4.90E-04	**15795.5**
f_7	50	7.95E-04	12326	50	5.09E-04	**717**
f_8	0	4.54E+00	500000	0	**1.25E-01**	500000
f_9	8	1.56E-03	500000	50	**9.21E-04**	**8338.5**
f_{10}	50	7.55E-04	17776	50	2.65E-04	**1060**
f_{11}	0	2.16E+00	500000	50	**5.75E-04**	**1650.5**
f_{12}	0	1.06E-01	500000	3	**3.69E-02**	500000
f_{13}	0	1.20E+01	500000	50	**6.10E-04**	**1986.5**
f_{14}	50	8.93E-04	20826	50	8.32E-04	**6008.5**
f_{15}	42	**4.87E-04**	**231150**	24	1.68E-03	500000

Table 3.4: Statistical comparison between standard Bees Algorithm and BA-HJ

Function	Standard Bees Algorithm	
	BA-HJ	
	Accuracy (*p*-value)	Speed (*p*-value)
f_1	1.0000	**0.0000**
f_2	1.0000	**0.0000**
f_3	1.0000	**0.0000**
f_4	1.0000	0.8808
f_5	1.0000	**0.0000**
f_6	1.0000	**0.0000**
f_7	1.0000	**0.0000**
f_8	**0.0000**	1.0000
f_9	**0.0000**	**0.0000**
f_{10}	1.0000	**0.0000**
f_{11}	**0.0000**	**0.0000**
f_{12}	**0.0000**	0.6101
f_{13}	**0.0000**	**0.0000**
f_{14}	1.0000	**0.0000**
f_{15}	**0.0000**	**0.0007**

In the low dimensional unimodal type benchmark functions such as *Martin Gaddy* (f_1) and *Booth* (f_2), the superiority of BA-HJ finding the global optimum is observed in Figures 3.2(a),(b). The addition of HJ in the local search seems to work very well in this type of landscape. The median speed of BA-HJ in both functions increased around four times compared to standard BA. Meanwhile, the performance of BA-HJ in low dimensional multimodal functions demonstrated some tremendous increase in the speed of the algorithm. Out of the four functions tested, three functions namely *Goldstein and Price* (f_3), *Six Hump Camel* (f_5) and *Michalewicz* (f_6) gave better results than standard BA. Figures 3.2(c),(e),(f) show the convergence graphs of the algorithms. However, in *Schaffer* (f_4), no significant improvement in the speed for both algorithms is observed. Nevertheless, as can be inferred from Figure 3.2(d), BA-HJ gave better fitness value at the beginning of the search compared to the standard BA due to the intensification of the local search but was insufficient to improve the final solution.

In high dimensional unimodal functions like *Hypersphere* (f_7), *Powell* (f_9) and *Axis* (f_{10}), BA-HJ excelled in terms of speed compared to standard BA but both algorithms failed to converge in *Rosenbrock* (f_8). However, there was an improvement in the accuracy of the solution for *Rosenbrock* (f_8) by BA-HJ. The median accuracy of the BA-HJ was closer to the global optimum compared to the standard BA as recorded in Table 3.3. Figures 3.2(g),(h),(i),(j) show the convergence graphs of the algorithms.

The set of benchmark functions used in this study comprised high dimensional multimodal functions such as *Ackley* (f_{11}), *Griewank* (f_{12}), *Rastrigin* (f_{13}), *Zakharov* (f_{14}) and *Styblinsky-Tang* (f_{15}). The convergence graphs of these functions are shown in Figures 3.2(k)-(o). The

BA-HJ outperformed the standard BA in four functions, *Ackley* (f_{11}), *Griewank* (f_{12}), *Rastrigin* (f_{13}), and *Zakharov* (f_{14}). However, the performance of BA-HJ in *Styblinsky-Tang* (f_{15}) was weaker than the standard BA. As can be inferred from Table 3.3, the BA-HJ success rate was only 48% compared to 84% by the standard BA and the *p*-value in Table 3.4 shows that the result was significant. This is maybe due to the further intensification of the local search using the HJ method that had trapped the algorithm in the local optima of the *Styblinsky-Tang* (f_{15}) and thus failed to converge.

Overall, the experimental results revealed that the standard BA has difficulty in dealing with some extremely complex multimodal functions such as *Michalewicz* (f_6), *Ackley* (f_{11}), *Griewank* (f_{12}), *Rastrigin* (f_{13}) and *Zakharov* (f_{14}). However, the success run, accuracy and convergence speed of the BA-HJ showed tremendous improvement in the case of these hard problems for the standard BA. This is because the pattern move in the HJ method can increase the number of directional searches of the standard BA. On the other hand, the global search mechanism in the BA-HJ can avoid premature convergence while approaching optimum by creating a large diversity of population.

Comparison was also made with SPSO2011 and qABC to gauge the performance of the BA-HJ. Table 3.5 presents the comparison results of the algorithms for the same set of benchmark functions mentioned earlier. The best solutions were highlighted in boldface. The same method of comparison was adopted by using the Mann-Whitney significance test and the results were recorded in Table 3.6. As can be inferred from Table 3.5, BA-HJ outclassed SPSO2011 and qABC in eleven (f_1, f_2, f_3, f_5, f_6, f_7, f_9, f_{10}, f_{11}, f_{13}, and f_{14}) out of fifteen functions. In the remaining four functions (f_4, f_8, f_{12} and f_{15}), BA-HJ performed better than SPSO2011 in terms of the

accuracy in two functions (f_8 and f_{15}). No improvement in performance was noted for function f_{12} while SPSO2011 came out as the winner for function f_4. Meanwhile, BA-HJ was better than qABC in functions f_4 and f_{12}. The results were comparable for functions f_8 and f_{15}.

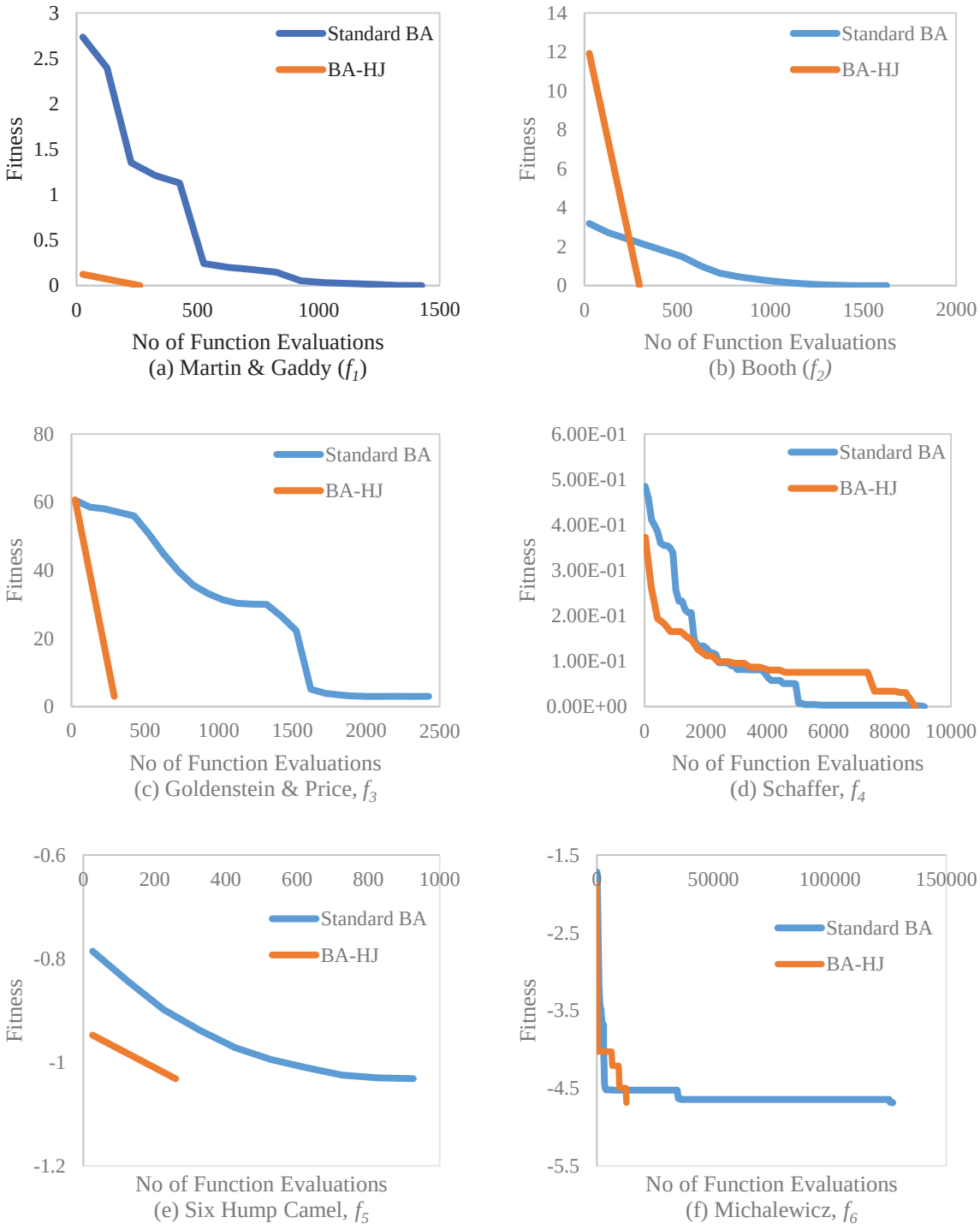


Figure 3.2: Sample graphs of convergence for standard Bees Algorithm and BA-HJ

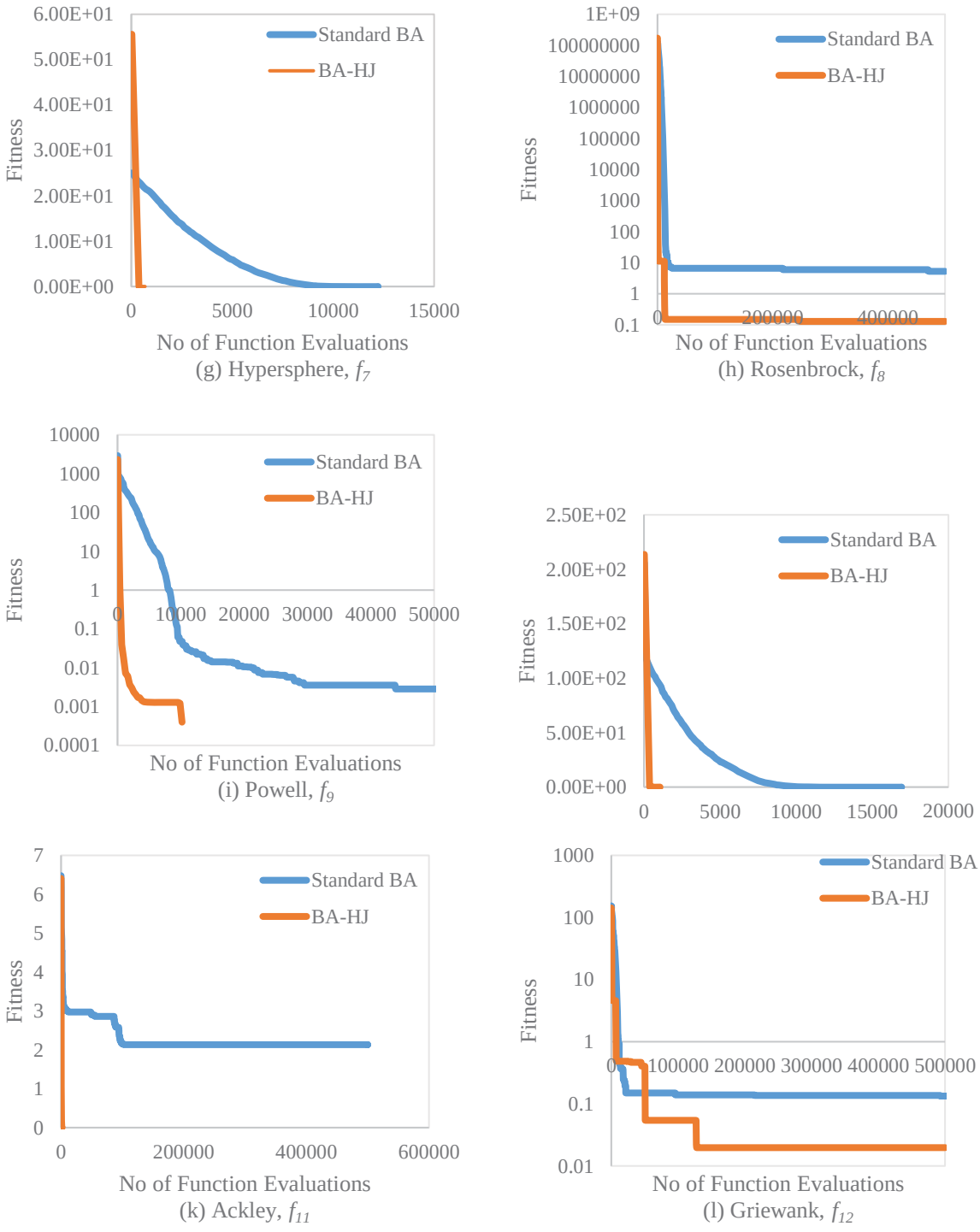


Figure 3.2: Sample graphs of convergence for standard Bees Algorithm and BA-HJ (continued)

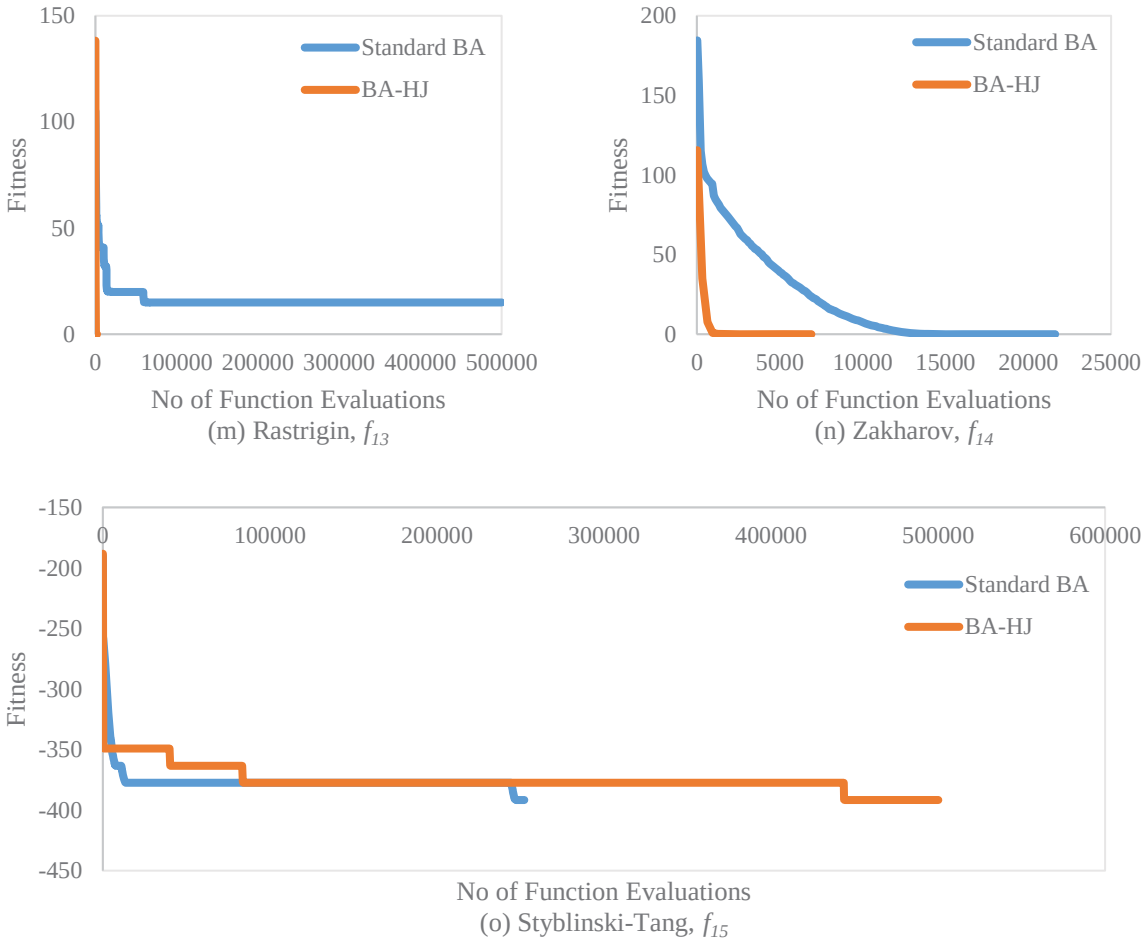


Figure 3.2: Sample graphs of convergence for standard Bees Algorithm and BA-HJ (continued)

To further compare the performance of the BA-HJ, standard BA, SPSO2011 and qABC in each benchmark functions, the results of 50 runs for all algorithms are shown in Figures 3.3 to 3.17. The speed performance was compared to see if many of the algorithms have found the optimum value whereas the accuracy performance was compared to assess if most of the algorithms have reached the maximum number of function evaluations. All figures clearly showed that BA-HJ produced a consistent result in all 50 runs for each function compared to others. This further validates the effectiveness of HJ in helping the search thus increasing the speed and accuracy

of the algorithm. The blue box in some of the figures represents the zoomed area of the graph for comparison purposes.

3.5 Engineering benchmark constrained and mechanical design problems

In this section, five well-studied constrained mechanical design problems are presented. These design problems have been solved and the best results obtained by the proposed BA-HJ have been compared with those produced by the standard BA and other algorithms reported in the literature. All the considered design problems have different kinds of objective functions, design variables, and constraints as shown in Table 3.7. These problems are discussed in detail in the following sub-sections and Appendix B. To evaluate the performance of the proposed BA-HJ for optimising the design problems, the parameter setting as in Table 3.8 was employed for both BA-HJ and the standard BA. Both algorithms were tested 30 times under the same environment using a computer with an Intel Xeon 2.40 GHz processor and 8GB of RAM. The maximum number of function evaluations was set to 30,000 for both algorithms.

Table 3.5: Performance comparison between BA-HJ, SPSO2011 and qABC

Function	BA-HJ			SPSO2011			qABC		
	Success	Accuracy	Speed	Success	Accuracy	Speed	Success	Accuracy	Speed
f_1	50	1.42E-08	**311**	50	5.42E-05	2800	38	8.82E-04	20223.5
f_2	50	5.48E-08	**287.5**	50	0.00E+00	3200	48	3.90E-04	2050
f_3	50	7.09E-06	**275**	50	5.00E-05	4000	50	2.16E-04	2450
f_4	50	1.54E-11	7584.5	50	0.00E+00	3500	32	7.55E-04	102706
f_5	50	2.84E-05	**271.5**	50	3.60E-05	3100	50	4.31E-04	950
f_6	50	4.90E-04	**15795.5**	25	3.20E-03	433650	50	4.71E-04	111436.5
f_7	50	5.09E-04	**717**	50	8.67E-05	10750	50	6.78E-04	92470
f_8	0	1.25E-01	500000	4	1.08E+01	500000	0	9.27E-02	500000
f_9	50	9.21E-04	**8338.5**	50	9.96E-05	88450	36	9.83E-04	269100.5
f_{10}	50	2.65E-04	**1060**	50	8.90E-05	13500	50	6.08E-04	117207.5
f_{11}	50	5.75E-04	**1650.5**	0	3.18E-01	500000	18	1.63E-03	500000
f_{12}	3	3.69E-02	500000	6	3.56E-02	500000	0	6.27E-02	500000
f_{13}	50	6.10E-04	**1986.5**	0	5.93E+00	500000	43	7.80E-04	372419.5
f_{14}	50	8.32E-04	**6008.5**	50	1.00E-04	44950	0	4.06E-02	500000
f_{15}	24	1.68E-03	500000	6	2.83E+01	500000	41	6.51E-04	144582

Table 3.6: Statistical comparison between BA-HJ, SPSO2011 and qABC

Function	BA-HJ			
	SPSO2011		qABC	
	Accuracy	Speed	Accuracy	Speed
f_1	1.0000	0.0000	1.0000	0.0000
f_2	1.0000	0.0000	1.0000	0.0000
f_3	1.0000	0.0000	1.0000	0.0000
f_4	1.0000	0.0000	1.0000	0.13104
f_5	1.0000	0.0000	1.0000	0.0000
f_6	1.0000	0.0000	1.0000	0.0000
f_7	1.0000	0.0000	1.0000	0.0000
f_8	0.0000	0.4902	0.07346	1.0000
f_9	1.0000	0.0000	1.0000	0.0000
f_{10}	1.0000	0.0000	1.0000	0.0000
f_{11}	0.0000	0.0000	0.0000	0.0000
f_{12}	0.62414	0.56192	0.0000	0.61006
f_{13}	0.0000	0.0000	0.01278	0.0000
f_{14}	0.0000	0.0000	0.0000	0.0000
f_{15}	0.0000	0.00988	0.0000	0.0027

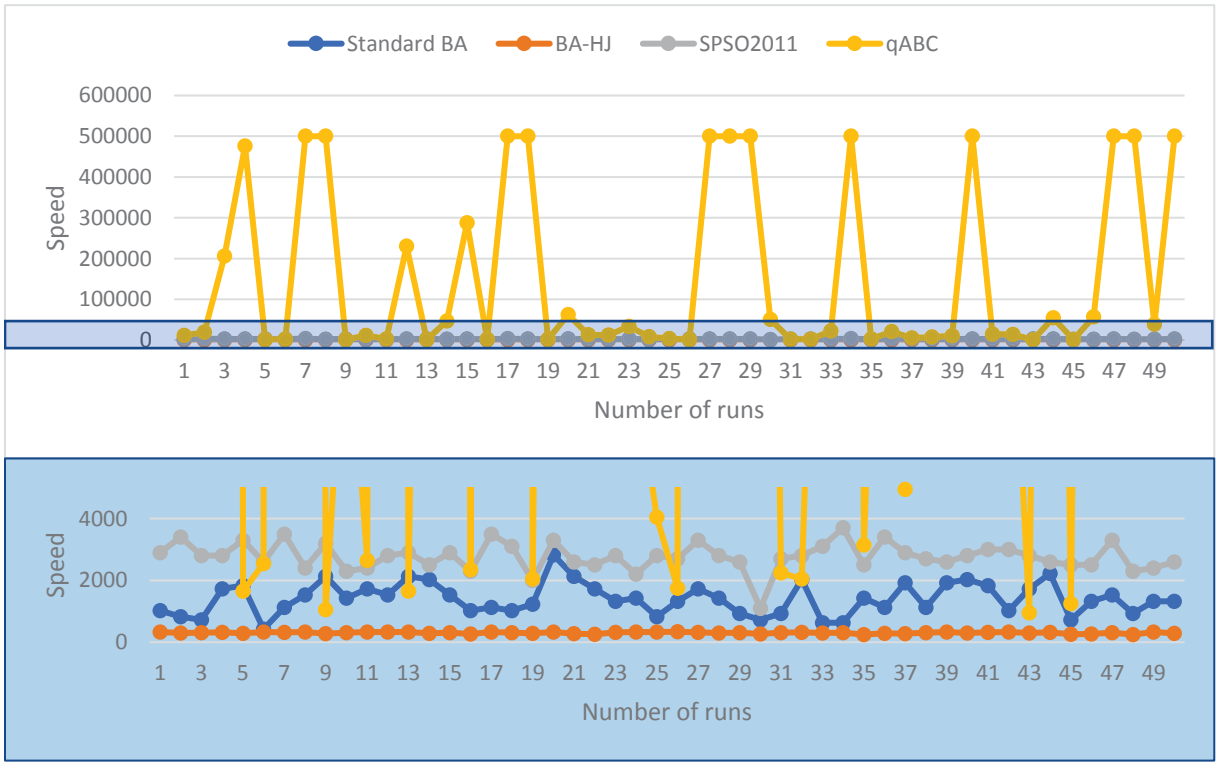


Figure 3.3: Comparison graph of speed performance between BA-HJ, SPSO2011 and qABC for f_1

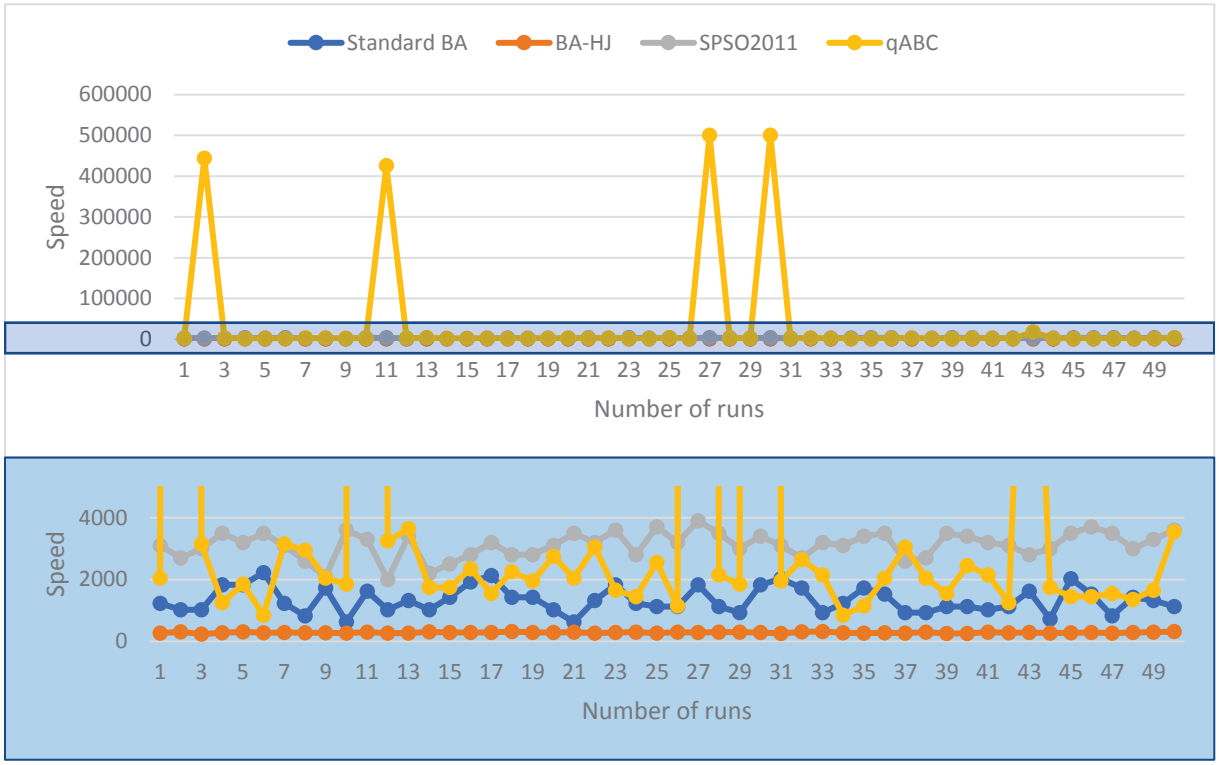


Figure 3.4: Comparison graph of speed performance between BA-HJ, SPSO2011 and qABC for f_2

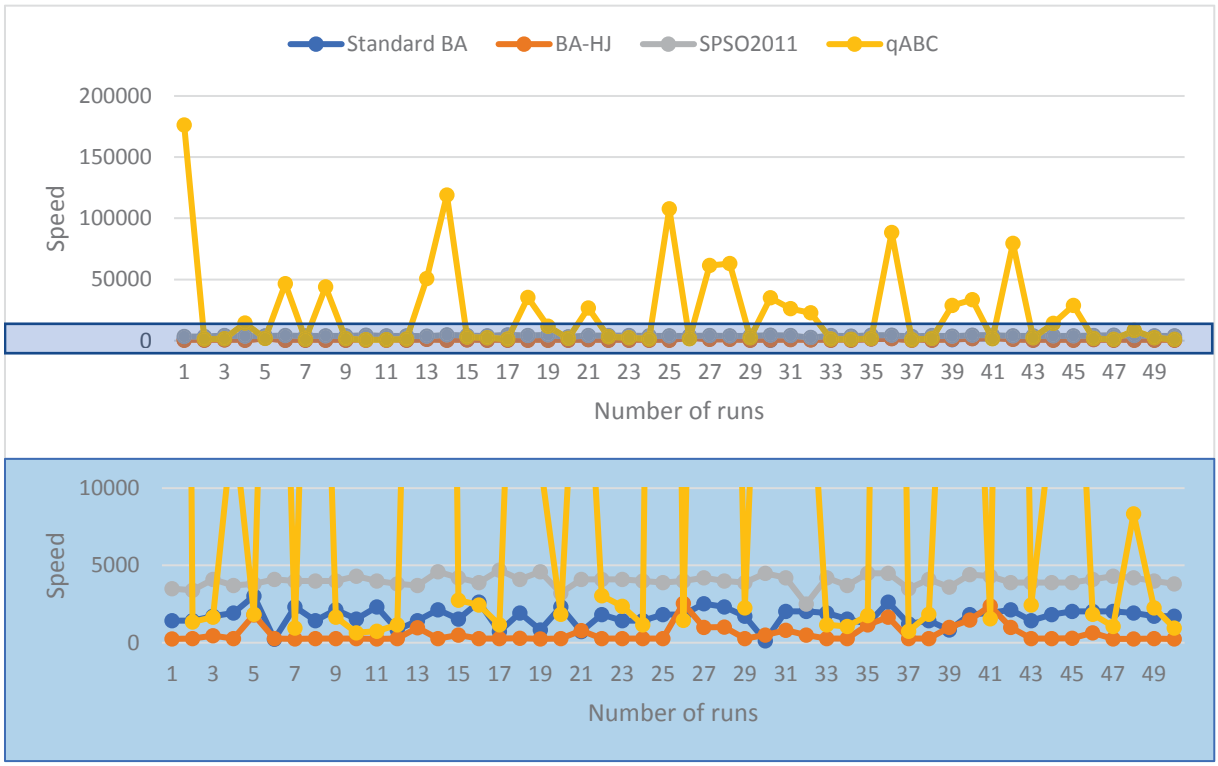


Figure 3.5: Comparison graph of speed performance between BA-HJ, SPSO2011 and qABC for f_3

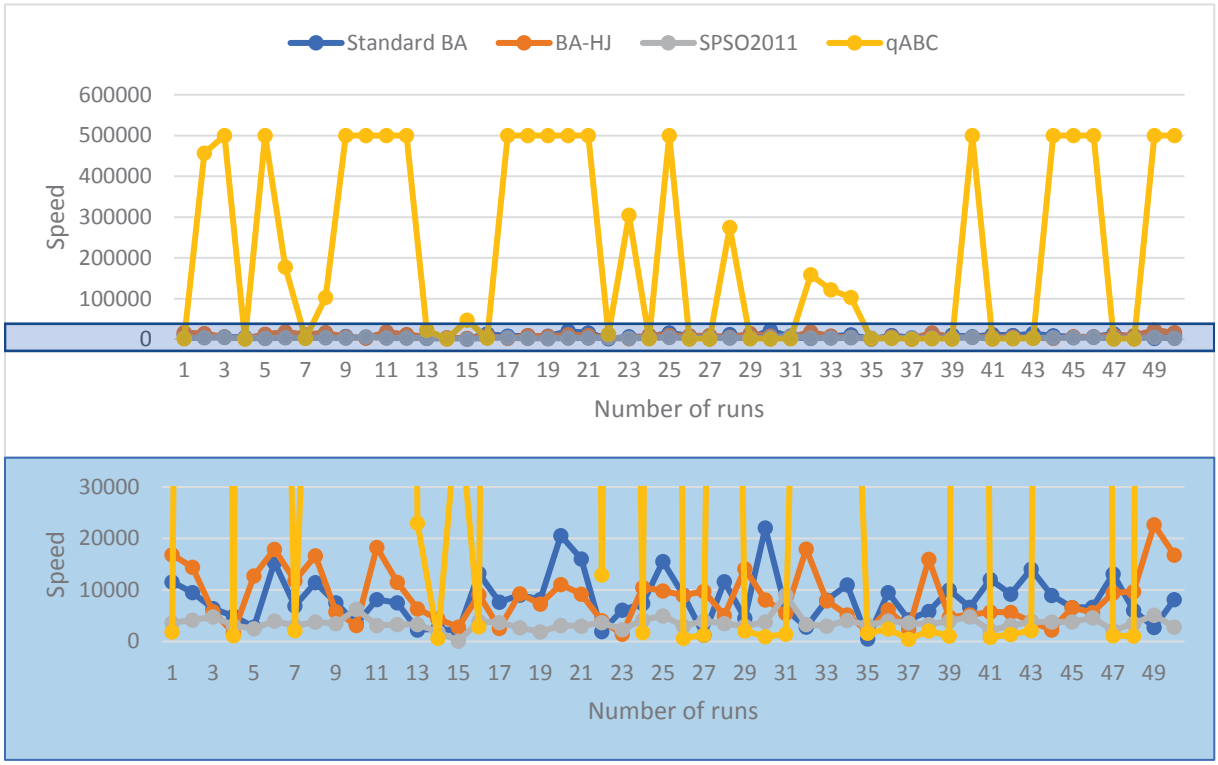


Figure 3.6: Comparison graph of speed performance between BA-HJ, SPSO2011 and qABC for f_4

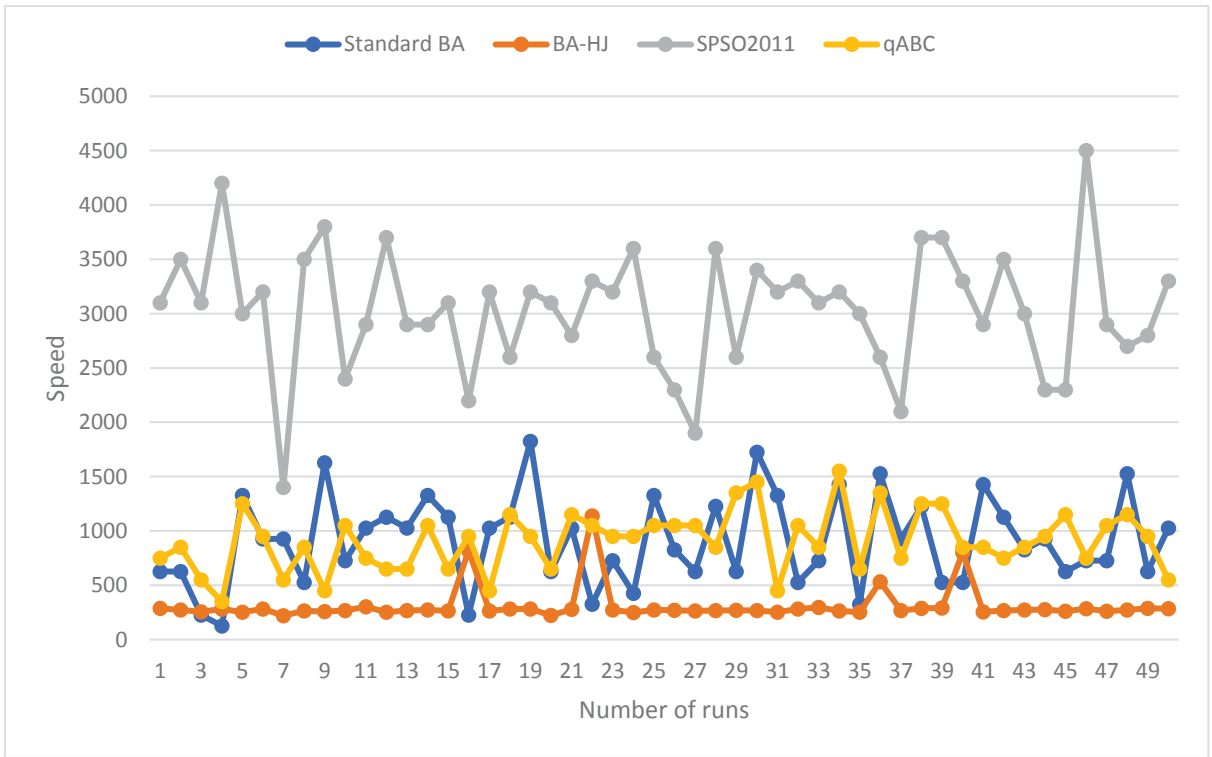


Figure 3.7: Comparison graph of speed performance between BA-HJ, SPSO2011 and qABC for f_5

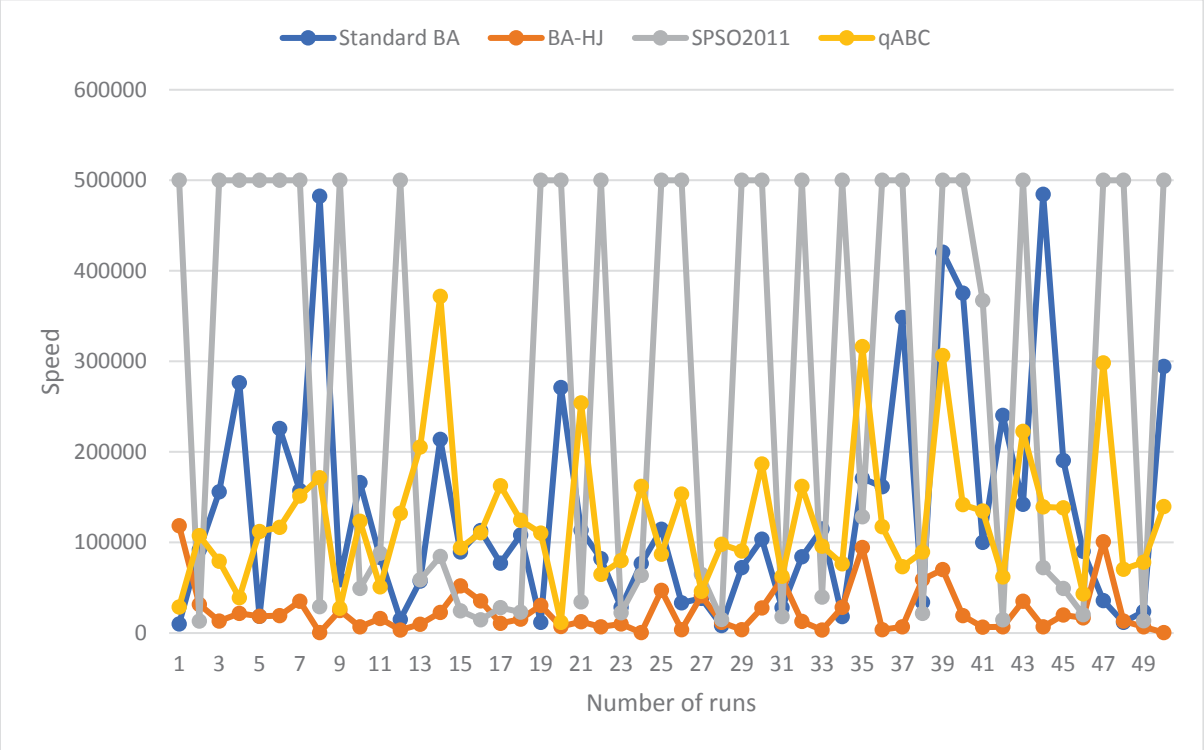


Figure 3.8: Comparison graph of speed performance between BA-HJ, SPSO2011 and qABC for f_6

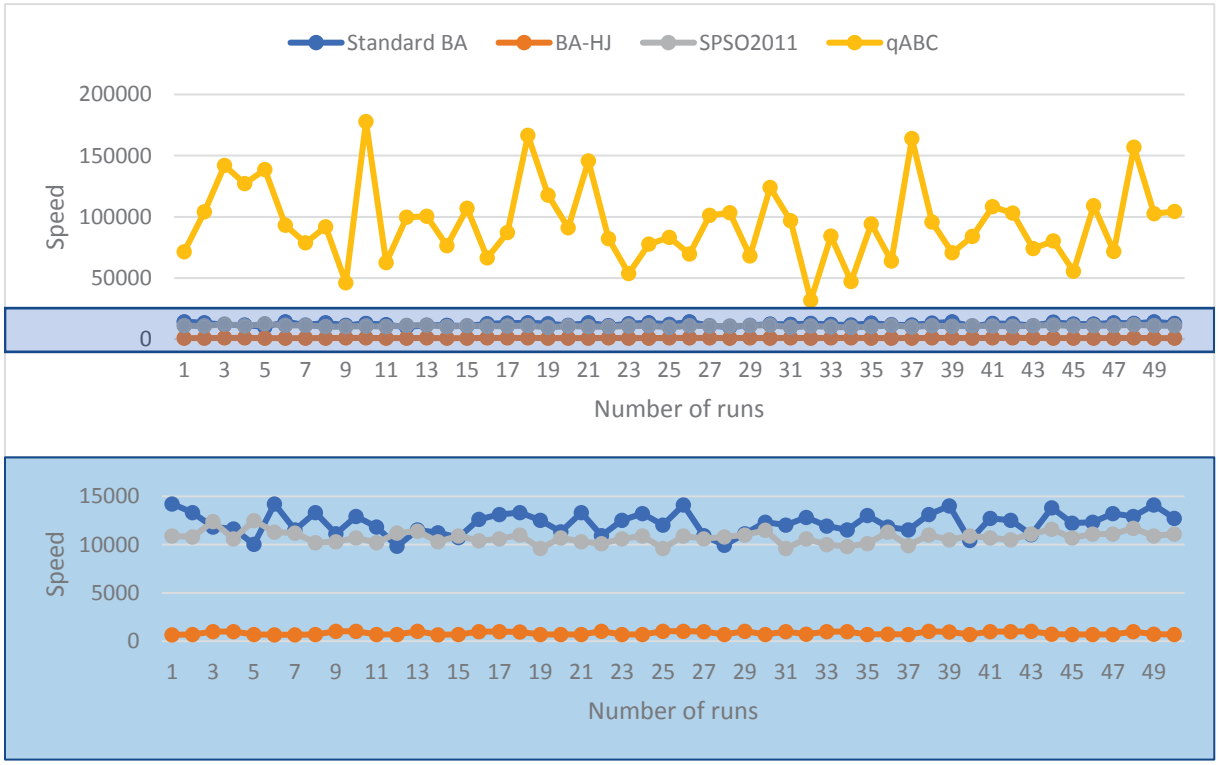


Figure 3.9: Comparison graph of speed performance between BA-HJ, SPSO2011 and qABC for f_7

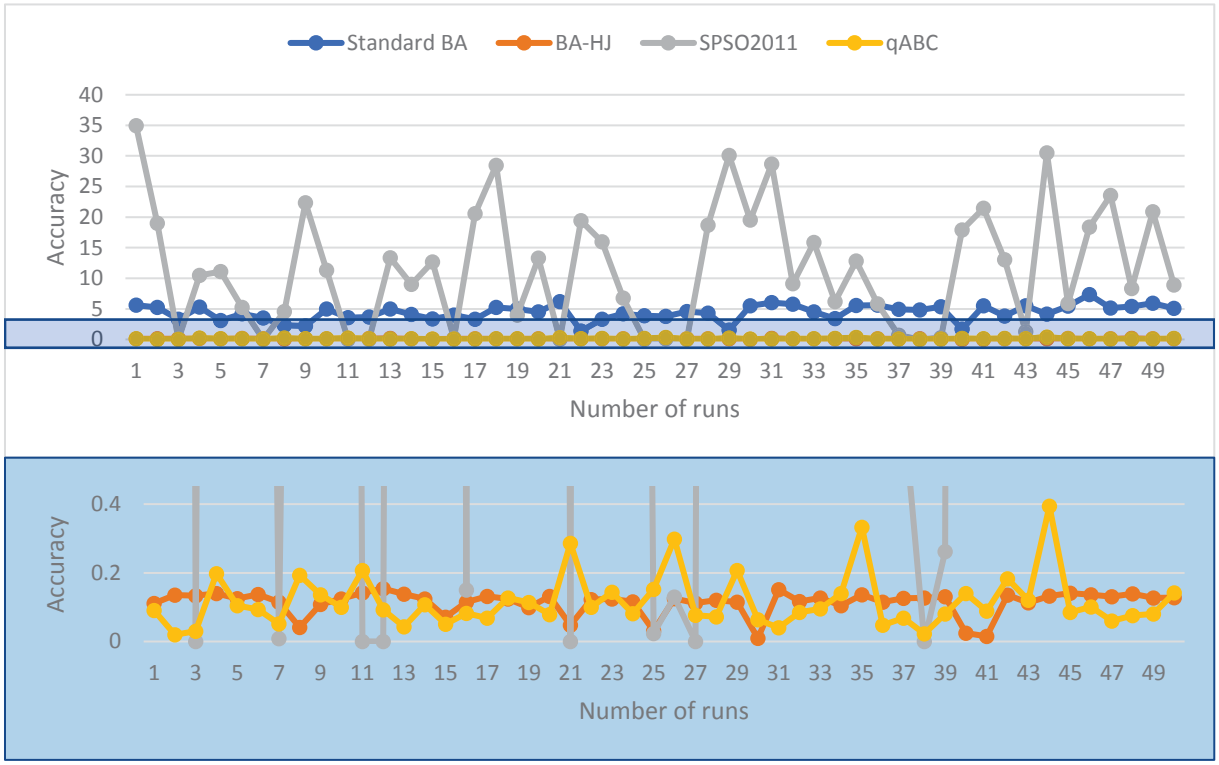


Figure 3.10: Comparison graph of accuracy performance between BA-HJ, SPSO2011 and qABC for f_8

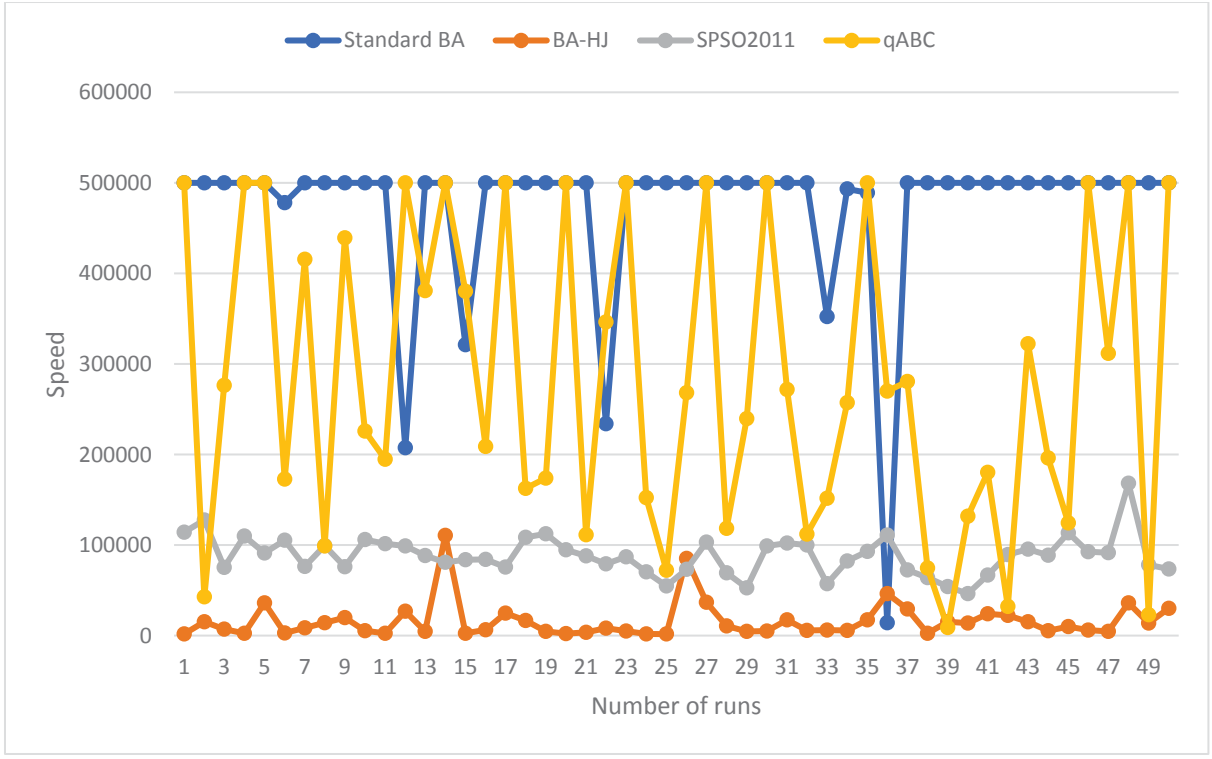


Figure 3.11: Comparison graph of speed performance between BA-HJ, SPSO2011 and qABC for f_9

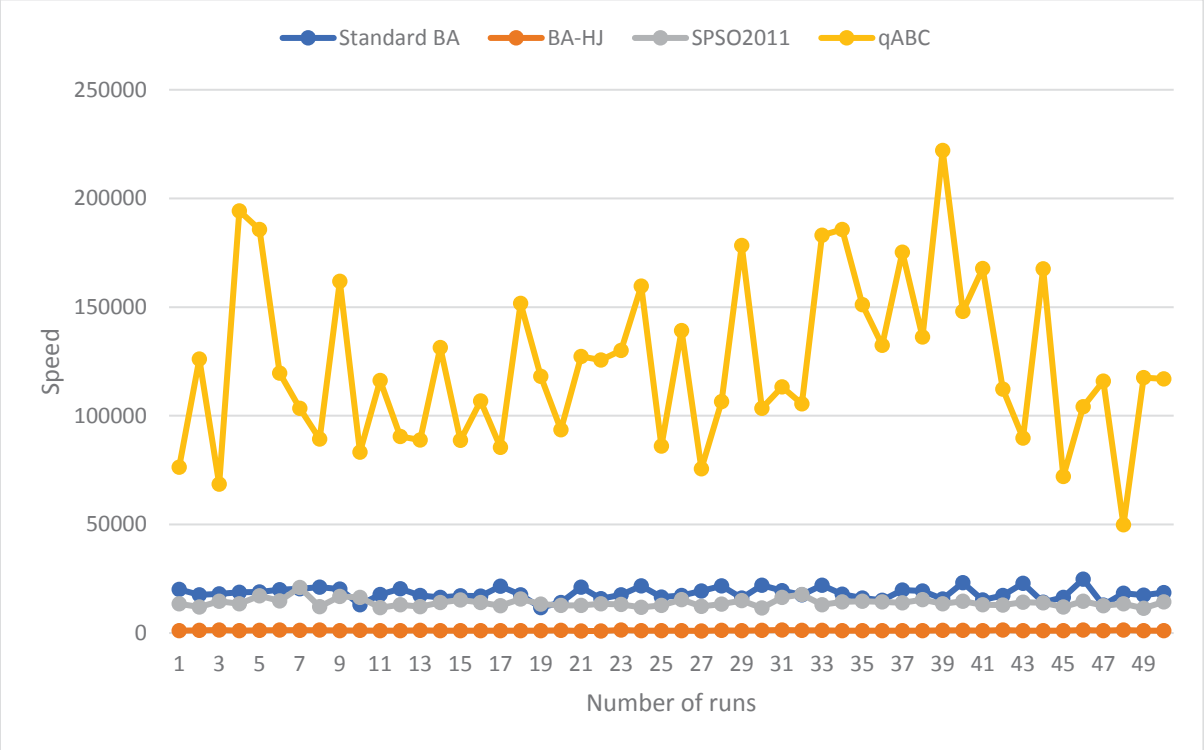


Figure 3.12: Comparison graph of speed performance between BA-HJ, SPSO2011 and qABC for f_{10}

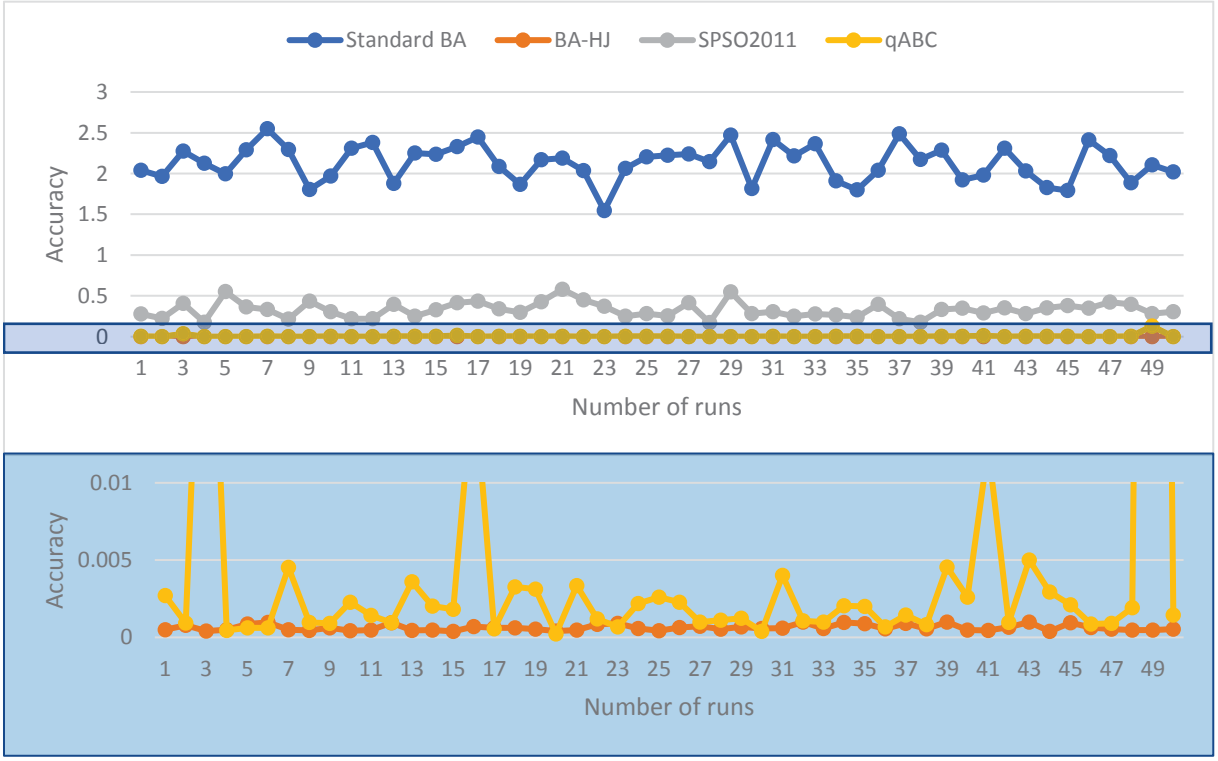


Figure 3.13: Comparison graph of accuracy performance between BA-HJ, SPSO2011 and qABC for f_{11}

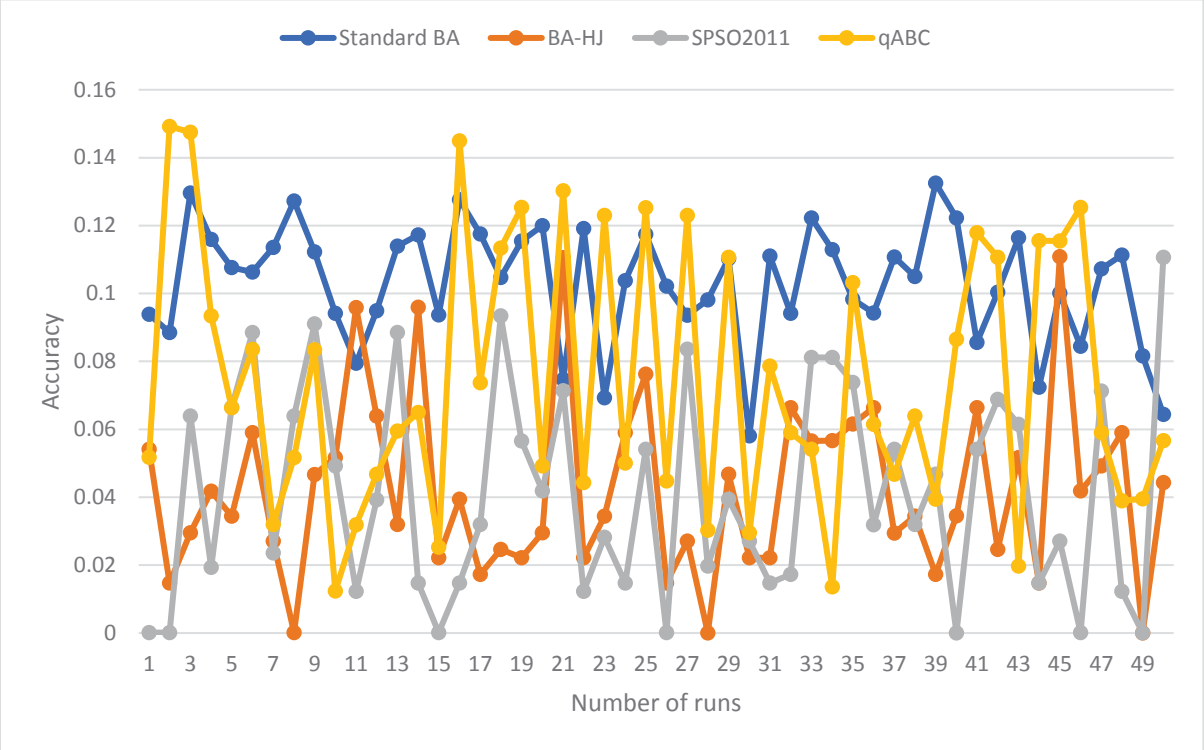


Figure 3.14: Comparison graph of accuracy performance between BA-HJ, SPSO2011 and qABC for f_{12}

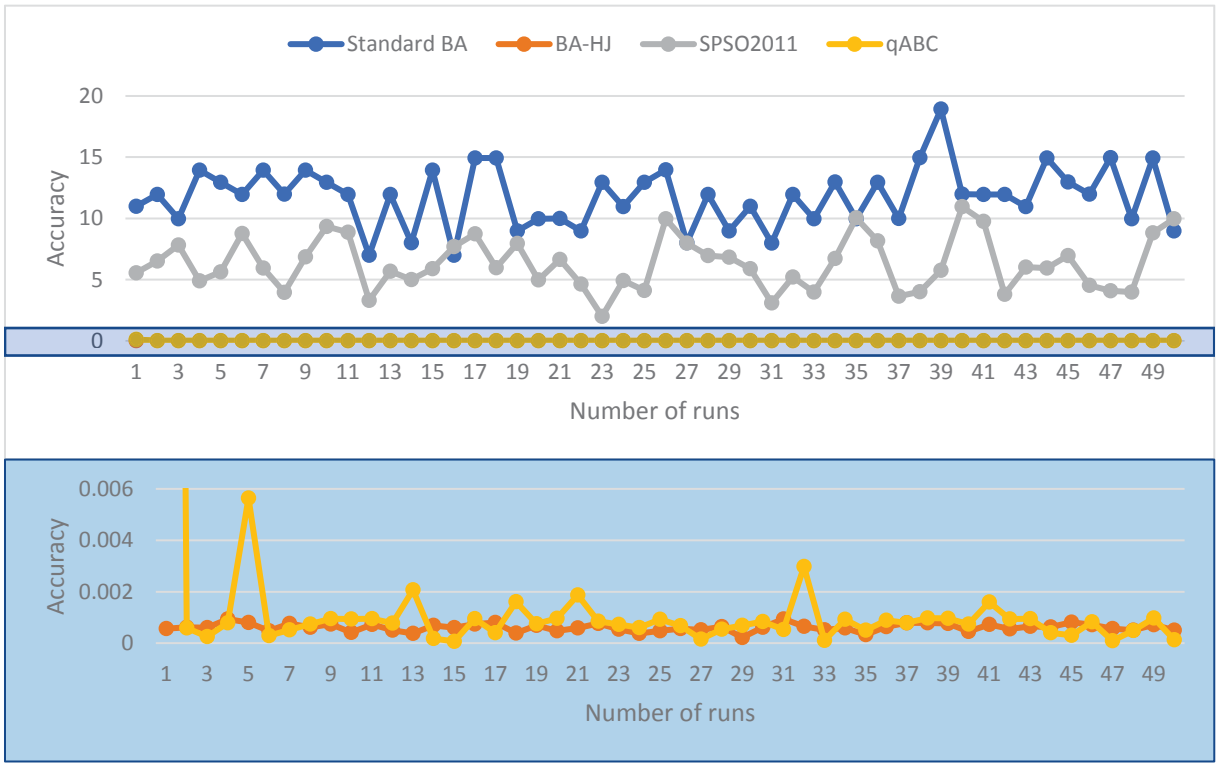


Figure 3.15: Comparison graph of accuracy performance between BA-HJ, SPSO2011 and qABC for f_{13}

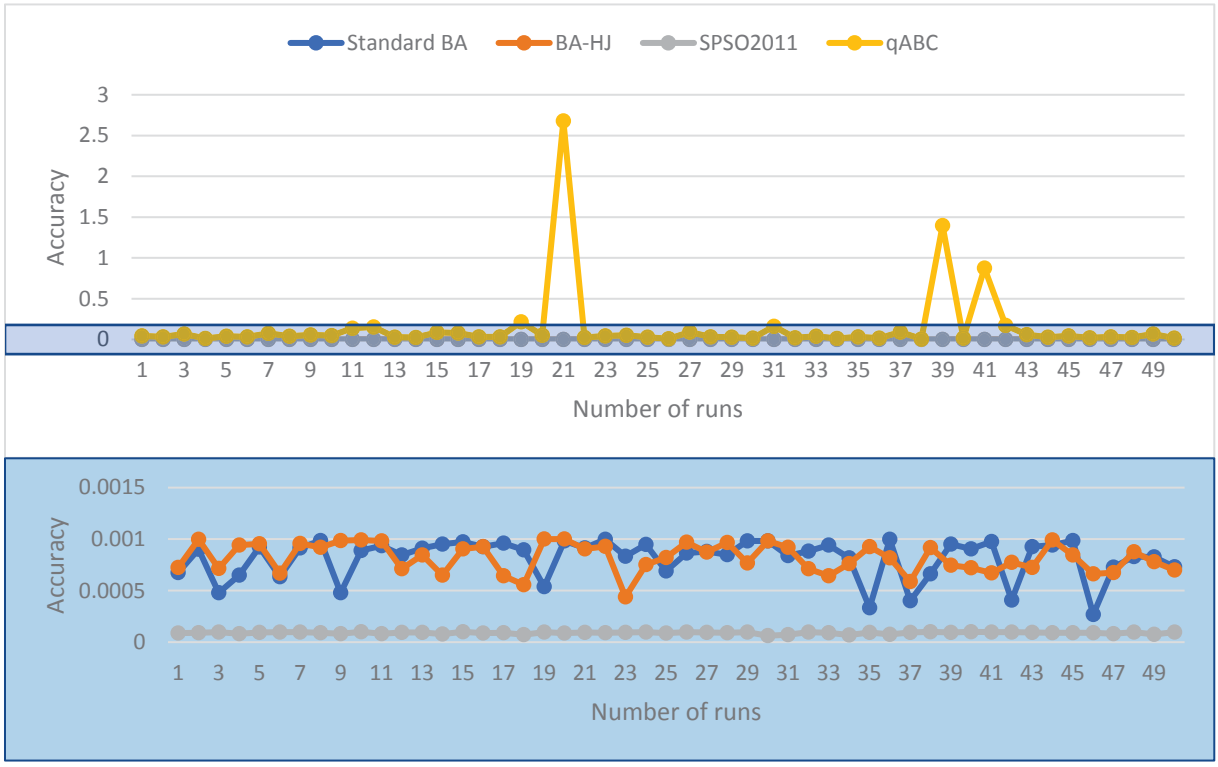


Figure 3.16: Comparison graph of accuracy performance between BA-HJ, SPSO2011 and qABC for f_{14}

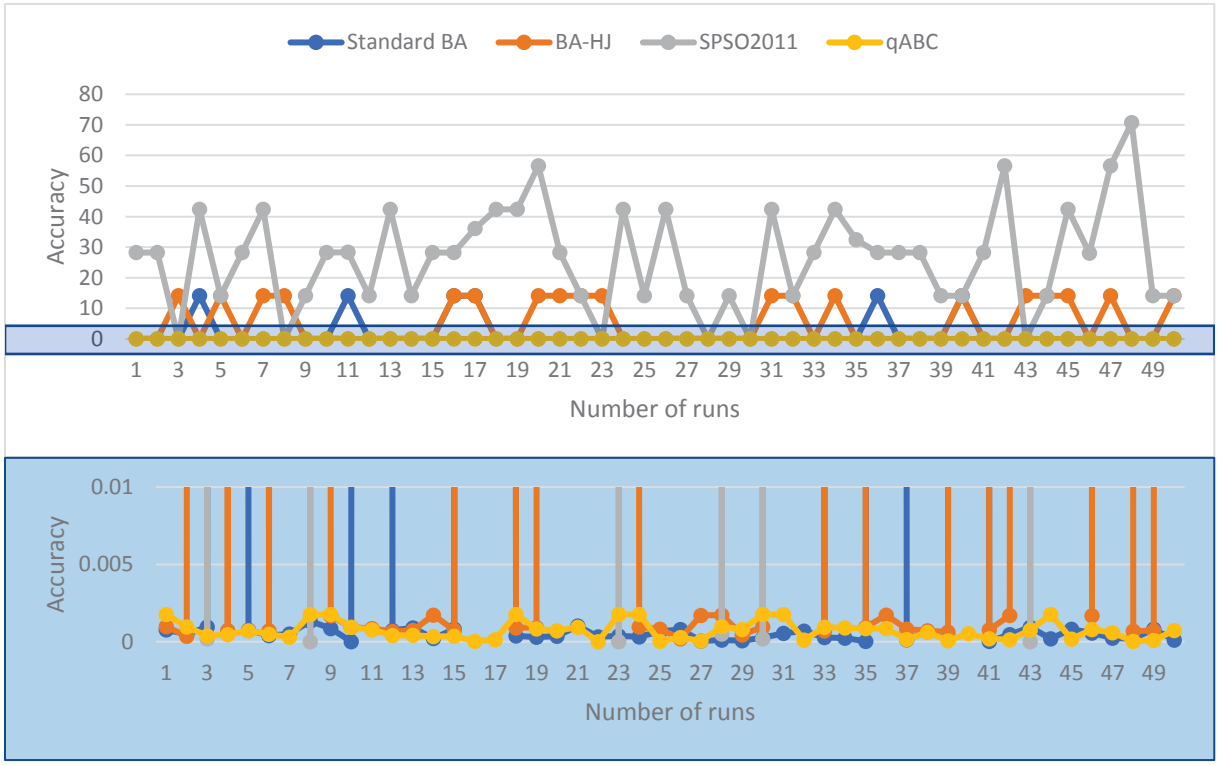


Figure 3.17: Comparison graph of accuracy performance between BA-HJ, SPSO2011 and qABC for f_{15}

Table 3.7: Constrained benchmark mechanical design problems

No	Problems	No. of design variables	Continuous design variables	Discrete design variables	No. of constraints	Objective
1	Three-bar truss	2	2	0	3	Minimise volume
2	Tension/compression spring	3	3	0	4	Minimise weight
3	Pressure vessel	4	2	2	4	Minimise cost
4	Welded beam	4	4	0	7	Minimise cost
5	Speed reducer	7	6	1	11	Minimise weight

Table 3.8: Parameter setting for constrained benchmark mechanical design problems

Parameter	**Value**
Number of scout bees, *ns*	10
Number of elite sites, *ne*	2
Number of best sites, *nb*	5
Recruited bees for elite sites, *nre*	8
Recruited bees for remaining best sites, *nrb*	5
Limit of stagnation cycles for site abandonment, *stlim*	5
Initial size of neighbourhood, *ngh*	
1. Three-truss bar	0.08
2. Tension/compression spring	0.003
3. Pressure vessel	1.0
4. Welded beam	0.08
5. Speed reducer	0.001

3.5.1 Three-bar truss design problem

The three-bar truss design problem is one of the engineering minimisation test problems for constrained algorithms. The objective of this design problem is to minimise the volume of a loaded three-bar truss subject to stress (r) constraints on each of the truss members by adjusting cross-sectional areas (x_1 and x_2) as shown in Figure 3.18. The formulation of the problem is shown in Appendix B.1.

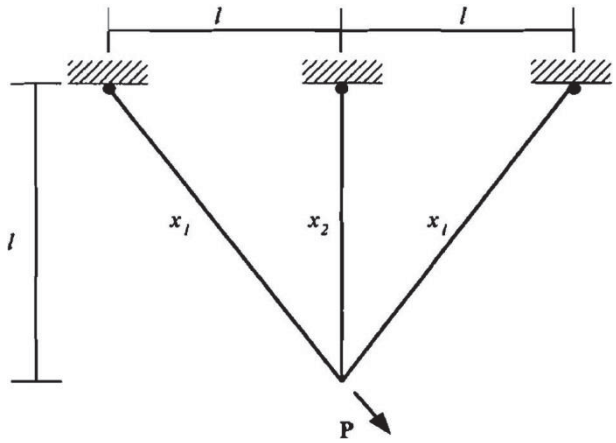


Figure 3.18: Schematic of three-bar truss design problem

3.5.2 Tension/compression spring design problem

This design optimisation problem as described by Arora (Arora, 1989) for which the objective is to minimise the weight of a tension/compression spring (Figure 3.19) subject to constraints on minimum deflection, shear stress, surge frequency, limits on outside diameter, and on design variables. The continuous independent variables are the wire diameter $d(x_1)$, the mean coil diameter $D(x_2)$, and the number of active coils $P(x_3)$. The formulation of the problem is presented in Appendix B.2.

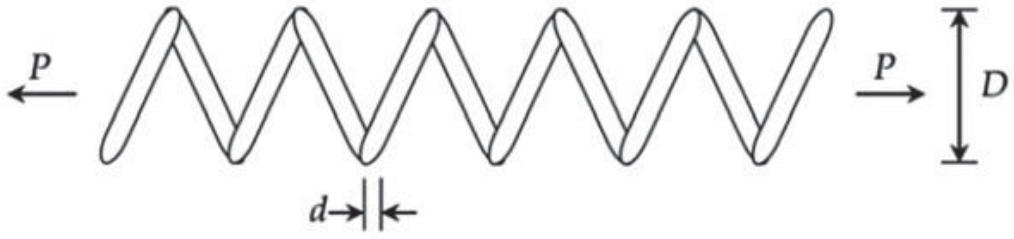


Figure 3.19: Tension/compression spring design problem

3.5.3 Pressure vessel design problem

In this constrained optimisation problem, proposed by Kannan and Kramer (Kannan & Kramer, 1994), the total cost of designing a pressure vessel is to be minimised (Figure 3.20). The total cost involves the cost of material, forming, and welding. The design variables for this optimisation problem are T_s (x_1, thickness of the shell), T_h(x_2, thickness of the head), R(x_3, inner radius), and L(x_4, length of the cylindrical section of the vessel). The design variables, T_s and T_h are expected to be integer multiples of 0.0625 in, and R and L are continuous variables. The mathematical formulation of the problem is presented in Appendix B.3.

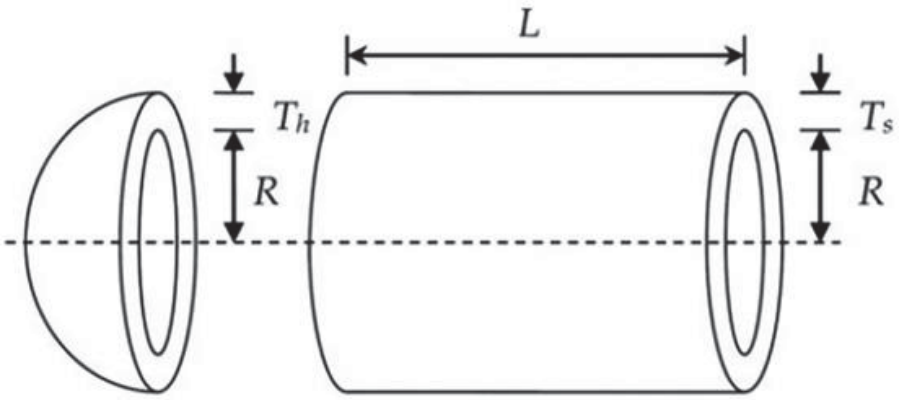


Figure 3.20: Pressure vessel design problem

3.5.4 Welded beam design problem

This well-known design problem (Figure 3.21) was proposed by Coello (Coello, 2000) aimed to minimise the cost of a welded beam design subject to constraints on shear stress (τ), bending stress (σ) in the beam, buckling load on the bar (P_b), end deflection of the beam (δ), and side constraints. There are four design variables as shown in Appendix B.4: $h(x_1)$, $l(x_2)$, $t(x_3)$ and $b(x_4)$.

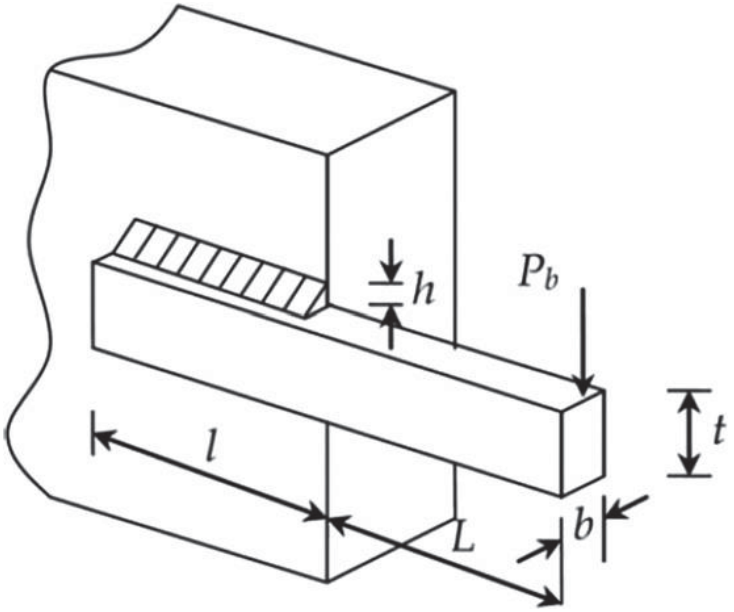


Figure 3.21: Welded beam design problem

3.5.5 Speed reducer design problem

The objective of this problem is to minimise the weight of the speed reducer (Figure 3.22) subject to constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts, and stresses in the shafts (Mezura-Montes & Coello, 2005). Variables x_1 to x_7 in this problem represent the face width (b), module of teeth (m), number of teeth in the pinion (z), length of the first shaft between bearings (l_1), length of the second shaft between bearings (l_2), and the diameter of first (d_1), and second shafts (d_2), respectively, as shown in Appendix B.5. The third variable x_3 (number of teeth in the pinion) is of integer values while all other variables are continuous.

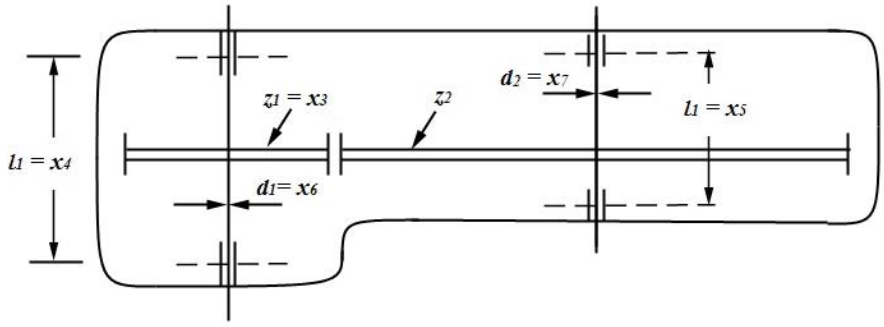


Figure 3.22: Speed reducer design problem

3.5.6 Comparison results of BA-HJ and standard BA

The performance of the newly developed algorithm, BA-HJ was compared to the standard BA on the five design problems presented in the previous sub-sections. Table 3.9 shows the best solutions for BA-HJ and the design variables obtained from the test experiments. The Mann-Whitney significant test was used on the results for 30 independent runs for all problems. To determine whether the difference between the medians was statistically significant, the *p*-values were compared to significance level of 0.05. The difference between the population medians was statistically significant if the *p*-value was less or equal to 0.05. Conversely, there was not enough evidence to conclude the difference between the population medians if the *p*-value was greater than 0.05.

The comparison results are shown in Table 3.10 and the best solution and statistically significant median are highlighted in boldface. The best solution found by BA-HJ was better than standard BA in all five benchmark mechanical design problems. In terms of median solutions, the BA-HJ was significantly superior than the standard BA in pressure vessel, welded beam, and speed reducer problems. Both algorithms performed equally in three-bar truss and tension/compression spring problems.

Table 3.9: Best results obtained by BA-HJ for constrained benchmark mechanical design problems

	Three-truss bar	**Tension/compression spring**	**Pressure vessel**	**Welded beam**	**Speed reducer**
$f(x)$	263.8962445	0.012666717	6066.19086	1.737432914	3005.542698
x_1	0.78891855	0.051854723	0.8125	0.203246944	3.500003345
x_2	0.407563819	0.360703654	0.4375	3.525229355	0.700000383
x_3		11.05981036	42.06142524	9.050190466	17
x_4			177.1799022	0.206611094	7.848867419
x_5					7.812147392
x_6					3.36489985
x_7					5.28713161

Table 3.10: Comparison of the statistical results obtained from BA-HJ and standard BA for constrained benchmark mechanical design problems

Problem		**Standard Bees Algorithm**	**BA-HJ**
Three-bar truss	Best	263.8964263	**263.8962445**
	Median	263.9015189	263.9015998
	Mean	263.9032913	263.9029512
	SD	0.005504271	0.005893028
	Worst	263.919459	263.9227632
	Evaluations	30,000	30,000
Tension/compression spring	Best	0.012670733	**0.012666717**
	Median	0.012681849	0.012682768
	Mean	0.012760301	0.012706748
	SD	0.00017804	5.53558E-05
	Worst	0.013402018	0.012871292
	Evaluations	30,000	30,000
Pressure vessel	Best	6119.246703	**6066.19086**
	Median	6222.419644	**6176.766071**
	Mean	6259.109338	6180.711514
	SD	126.2859774	65.45648595
	Worst	6522.111587	6294.84658
	Evaluations	30,000	30,000
Welded beam	Best	1.73958454	**1.737432914**
	Median	1.767187634	**1.755177218**
	Mean	1.766361958	1.758962845
	SD	0.0150332	0.0137442
	Worst	1.810428812	1.788860218
	Evaluations	30,000	30,000
Speed reducer	Best	3006.240405	**3005.542698**
	Median	3038.251131	**3031.239948**
	Mean	3035.608189	3029.411968
	SD	10.27826125	9.650873923
	Worst	3050.226338	3046.51389
	Evaluations	30,000	30,000

3.5.7 Comparison results with algorithms reported in the literature

The BA-HJ performance in the five design problems was further compared to other algorithms available in the literature. In the three-truss bar design problem, their performance was compared to the Society and Civilisation algorithm (SC) (Ray & Liew, 2003) and Particle Swarm with Differential Evolution (PSO-DE) (Liu, Cai, & Wang, 2010). For tension/compression spring, pressure vessel, and welded beam optimisation problem, comparison were made with genetic algorithm with co-evolution adaptation (GA1) (Coello, 2000), dominance-based selection scheme for genetic algorithm (GA2) (Coello & Montes, 2002), unified particle swarm optimisation (UPSOm) (Parsopoulos & Vrahatis, 2005), artificial bee colony algorithm (ABC) (Akay & Karaboga, 2012), and PSO-DE. The chosen algorithms for speed reducer optimisation problem are SC, PSO-DE, and ABC. The statistical results are presented in Tables 3.11 to 3.15.

As observed from Tables 3.11 through 3.15, the BA-HJ produced generally comparable results in comparison with other algorithms. Even though the comparison results showed that PSO-DE found the best solution in three-truss bar and pressure vessel, ABC found the best solution in tension/compression spring and welded beam, and SC found the best solution in speed reducer, these results are comparatively close to the results of BA-HJ.

Table 3.11: Comparison results for the three-truss bar optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
SC	263.895846	263.903356	263.969756	17,610
PSO-DE	263.89584338	263.89584338	263.89584338	17,600
Bees Algorithm	263.8964263	263.9032913	263.919459	30,000
BA-HJ	263.8962445	263.9029512	263.9227632	30,000

Table 3.12: BA-HJ comparison results for the tension/compression spring optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
GA1	0.0127048	0.0127690	0.0128220	900,000
GA2	0.0126810	0.0127420	0.0129730	80,000
PSO-DE	0.012665233	0.012665233	0.012665233	42,100
UPSOm	0.0131200	0.0229478	-	100,000
ABC	0.012665	0.012709	-	30,000
Bees Algorithm	0.012670733	0.012760301	0.013402018	30,000
BA-HJ	0.012666717	0.012706748	0.012871292	30,000

Table 3.13: BA-HJ comparison results for the pressure vessel optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
GA1	6288.7445	6293.8432	6308.4970	900,000
GA2	6059.9463	6177.2533	6469.3220	80,000
PSO-DE	6059.714335	6059.714335	6059.714335	42,100
UPSOm	6544.27	9032.55	-	100,000
ABC	6059.714736	6245.308144	-	30,000
Bees Algorithm	6119.246703	6259.109338	6522.111587	30,000
BA-HJ	6066.19086	6180.711514	6294.84658	30,000

Table 3.14: BA-HJ comparison results for the welded beam optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
GA1	1.748309	1.771973	1.785835	900,000
GA2	1.728226	1.792654	1.993408	80,000
PSO-DE	1.724852309	1.724852309	1.724852309	66,600
UPSOm	1.92199	2.83721	-	100,000
ABC	1.724852	1.741913	-	30,000
Bees Algorithm	1.73958454	1.766361958	1.810428812	30,000
BA-HJ	1.737432914	1.758962845	1.788860218	30,000

Table 3.15: BA-HJ comparison results for the speed reducer optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
SC	2994.744241	3001.758264	3009.964736	54,456
PSO-DE	2996.348165	2996.348165	2996.348166	70,100
ABC	2997.058412	2997.058412	-	30,000
Bees Algorithm	3006.240405	3035.608189	3050.226338	30,000
BA-HJ	3005.542698	3029.411968	3046.51389	30,000

3.6 Summary

This chapter presented the modified version of the Bees Algorithm called BA-HJ. It was aimed at enhancing the convergence speed by intensifying the local search phase of the algorithm using the HJ method. The proposed BA-HJ was tested on a set of unconstrained benchmark functions and has showed a very reliable performance in most cases. Then, when compared with the standard BA and two well-known swarm-based algorithms, the proposed BA-HJ demonstrated strong competitive results in terms of its success at locating the optimum solution, convergence speed, and accuracy. The pattern move element in the BA-HJ helped to increase the directional search in the local search and at the same time maintained the population diversity through its unique global search mechanism.

Furthermore, the effectiveness of HJ in guiding local search in BA-HJ was shown in the results of constrained mechanical design optimisation problems. The proposed BA-HJ produced the best solution in all five design problems tested compared to standard BA. Finally, when compared to other algorithms reported in the literature, the proposed algorithm had shown comparable performance in terms of best solution found.

CHAPTER 4

BEES ALGORITHM WITH CHAOS (CHAOSBA)

4.1 Preliminaries

Randomisation has been used by many algorithms in the local or global search procedure including the standard BA. This method is practised in the deployment of bees for the standard BA exploitation and exploration phases. Recently, chaotic sequences have been successfully adopted to replace the random method to enrich the search behaviour and to avoid getting trapped at local optima (Caponetto et al., 2003). In this chapter, the chaos method was used to propose the Bees Algorithm with Chaos (ChaosBA) and the performance of the algorithm was tested on several benchmark functions and applications.

The rest of the chapter is organised as follows. Section 4.2 presents the proposed ChaosBA, Section 4.3 and Section 4.4 respectively explain the experimental setup and the results obtained for unconstrained function optimisation; Section 4.5 presents the experimental setup and results for the selected constrained mechanical engineering benchmark problems; Section 4.6 concludes the chapter.

4.2 Bees Algorithm with Chaos (ChaosBA)

The inspiration behind the development of ChaosBA was based on the idea of replacing the random method currently used in standard BA. Previous studies have shown improvements in

the performance of other algorithms using chaos sequences (Gandomi et al., 2013; Gandomi & Yang, 2014; Gholipour, Khosravi, & Mojallali, 2015; Lewis, 2014; Liu et al., 2005). Based on the consideration to take advantage of the well-known characteristics of the chaotic systems, ChaosBA was developed to enrich and increase the exploitation power of the standard BA in the quest to improve the performance of the algorithm.

Chaos theory is coined by Lorenz (Lorenz, 1963) as the so-called 'butterfly effect' in his attempt to simulate the global weather system. He discovered that minute changes in the initial conditions directed the resulting simulations towards drastically different final results. In general, chaos is a classical nonlinear system characterised by ergodicity, randomicity, and sensitivity to its initial conditions (Koupaei, Hosseini, & Ghaini, 2016; Li & Jiang, 1998). Due to these unique characteristics, chaotic sequences generated from the well-known logistic map were used in ChaosBA. The logistic map equation employed in the proposed algorithm is as follows:

$$x_{n+1} = f(\mu, x_n) = \mu x_n(1 - x_n) \tag{15.1}$$

where μ is a control parameter, n = 0,1,2,…, and x is a variable. Suppose $0 < x_0 < 0$, $0 \leq \mu \leq 4$. It is easy to find that Eq. (4.1) is a deterministic system without any stochastic disturbance. It seems that its long-time behaviour can be predicted. But that is not true. The behaviour of system Eq. (4.1) is greatly changed with the variation of μ. When μ = 4, Eq. (4.1) behaves chaotically in an unpredictable pattern, the equation is changed to:

$$x_{n+1} = 4x_n(1 - x_n) \tag{4.2}$$

As seen in Figure 4.1, the logistic map presents significantly different dynamics, periodic sequences in $\mu \leq 3.71$ and chaotic sequences in $3.71 \leq \mu \leq 4$. Very small changes in the initial

value of μ will cause a large difference in its long-term behaviour, a typical chaotic characteristic.

The ChaosBA proposed in this chapter uses the chaotic sequences from Eq. (4.2) to generate new candidate solutions in the elite sites, best sites, and global search procedures. In the standard BA, after the elite and best sites are located, the process of producing new candidate solutions in the local search by sending out recruited bees are randomly generated. Similarly, the remaining bees in the global search procedure are also randomly produced in the search space. The procedures of ChaosBA is illustrated in the flowchart in Figure 4.2.

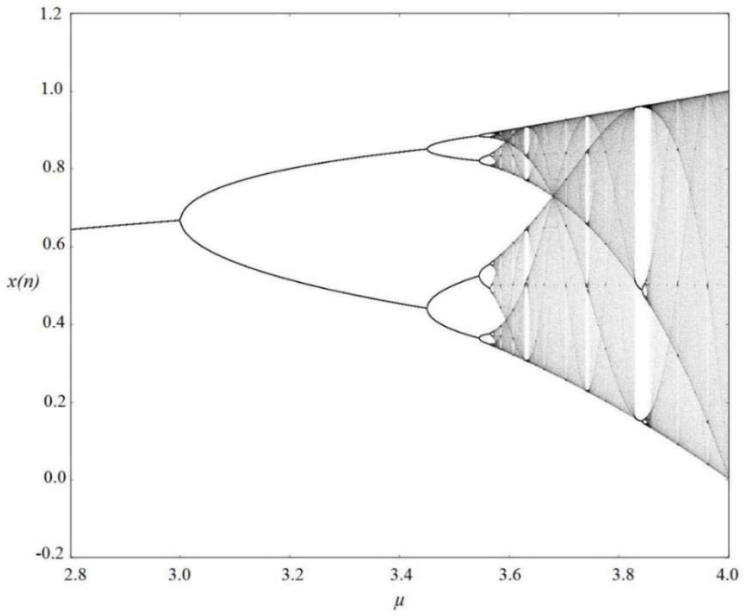


Figure 4.1: Bifurcation diagram of logistic map

ChaosBA begins with initialising the whole population as the scout bees. The idea was to get as many points as possible to cover the search space. In this case, the population was set to 100, therefore, the scout bees parameter was set to 100 for initialisation. According to the flowchart in Figure 4.2, ChaosBA continues with the fitness evaluations procedure to determine the elite and best sites for the local search procedure to take place. All procedures up to this phase are

similar to standard BA procedures. Once the elite and best sites are selected, the deployment of recruited bees for both sites is made by using chaotic sequences from Eq. (4.2).

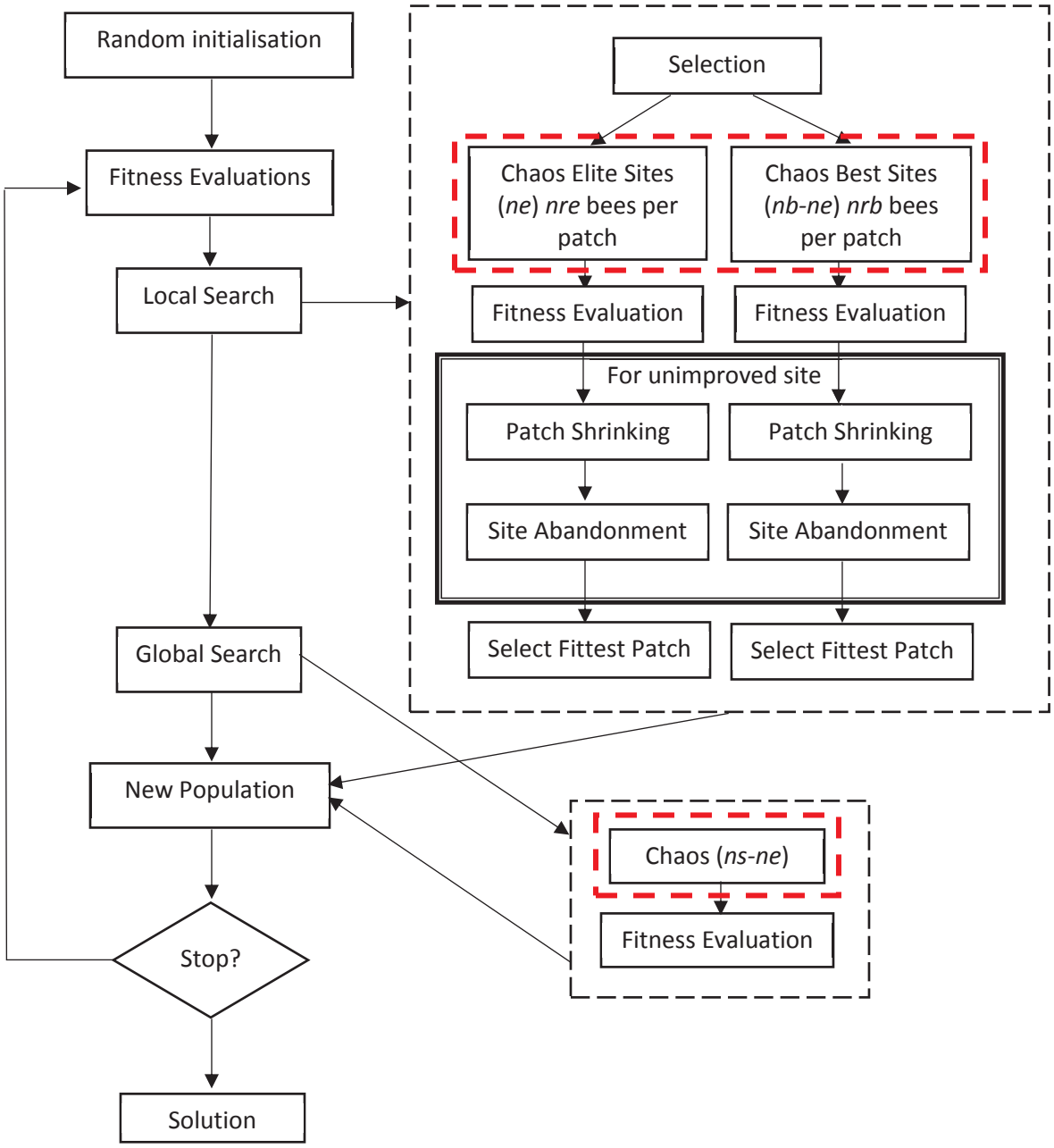


Figure 4.2: Flow chart of Bees Algorithm with chaos (ChaosBA)

The points of the current elite and best sites were utilised as the initial points for the chaotic sequences to start with. After that, ChaosBA undergoes similar procedures as the standard BA

to select the fittest patches for each site. For the global search procedure, ChaosBA uses Eq. (4.2) to generate chaotic sequences using the domains of the problem as the upper and lower limits. Finally, the points gathered from the local search (elite and best sites) and global search are put together in the newly improved population.

4.3 Experimental Setup

In general, the experimental setup in this chapter followed the setup as in Section 3.3. Moreover, this chapter used similar benchmark functions as in Appendix A and parameters setting in Table 3.1, as well as the stopping criteria described in Section 3.3. The performance of the proposed algorithm was compared to similar as algorithms in Section 3.3.

4.4 Results and Discussion

Likewise, the performance of ChaosBA was compared to the standard BA to measure the improvement of the algorithm. Table 4.1 presents the comparison between these two algorithms in terms of success runs, accuracy compared to the known optimum value, and convergence speed. The performance of the algorithm that is significant over the other is written in boldface. The Mann-Whitney significance test was used and the *p*-values for both algorithms are tabulated in Table 4.2. If both algorithms found the known optimum value of the function, the speed of the algorithm to get there was compared. Since the speed used in this research was the number of function evaluation being called during the search, the lesser speed indicates the better the algorithm for that function.

As seen in Table 4.1, there is a significant improvement in the results of the ChaosBA in the majority of the functions. Altogether, ChaosBA outperformed the standard BA in twelve out of fifteen benchmark functions. In the other three benchmark functions, the ChaosBA and standard BA produced a comparable performance. In addition, Figure 4.3 shows the sample graphs of convergence for both algorithms in all fifteen benchmark functions. As seen in some of the sample graphs such as *Booth*, *Goldenstein & Price*, *Six Hump Camel*, *Rastrigin*, *Ackley* and *Zakharov*, the ChaosBA managed to reach the optimum faster even though they started the search with higher fitness value than the standard BA. The simulation results above showed that the newly proposed optimisation method, ChaosBA has successfully improved the performance of the standard BA by utilising the three basic traits of chaotic variables, namely pseudo-randomness, ergodicity, and irregularity in its search. The addition of these unique characteristics showed that the capability and applicability of the proposed method were fully illustrated through all the benchmark functions simulation results.

Table 4.1: Performance comparison between standard Bees Algorithm and ChaosBA

Function	**Standard Bees Algorithm**			**ChaosBA**		
	Success	**Accuracy**	**Speed**	**Success**	**Accuracy**	**Speed**
f_1	50	3.90E-04	1376	50	3.99E-04	**900**
f_2	50	3.81E-04	1226	50	4.21E-04	**1000**
f_3	50	5.06E-04	1826	50	4.27E-04	1802.5
f_4	50	2.39E-04	7583.5	50	1.63E-04	**2552**
f_5	50	3.90E-04	926	50	2.83E-04	**650**
f_6	50	5.20E-04	96096.5	23	4.12E-02	500000
f_7	50	7.95E-04	12326	50	8.11E-04	**5058**
f_8	0	4.54E+00	500000	0	**2.75E+00**	500000
f_9	8	1.56E-03	500000	50	**7.74E-04**	**38294**
f_{10}	50	7.55E-04	17776	50	8.29E-04	**10274**
f_{11}	0	2.16E+00	500000	0	**7.26E-02**	500000
f_{12}	0	1.06E-01	500000	0	1.01E-01	500000
f_{13}	0	1.20E+01	500000	0	**1.06E-02**	500000
f_{14}	50	8.93E-04	20826	50	7.90E-04	**11881**
f_{15}	42	4.87E-04	231150	50	6.03E-04	**10272**

Table 4.2: Statistical comparison between standard Bees Algorithm and ChaosBA

Function	Standard Bees Algorithm ChaosBA	
	Accuracy (p-value)	**Speed (p-value)**
f_1	1.0000	**0.0010**
f_2	1.0000	**0.0018**
f_3	1.0000	0.1310
f_4	1.0000	**0.0000**
f_5	1.0000	**0.0002**
f_6	**0.0000**	**0.0000**
f_7	1.0000	**0.0000**
f_8	**0.0000**	1.0000
f_9	**0.0000**	**0.0000**
f_{10}	1.0000	**0.0000**
f_{11}	**0.0000**	1.0000
f_{12}	0.6892	1.0000
f_{13}	**0.0000**	1.0000
f_{14}	1.0000	**0.0001**
f_{15}	1.0000	**0.0000**

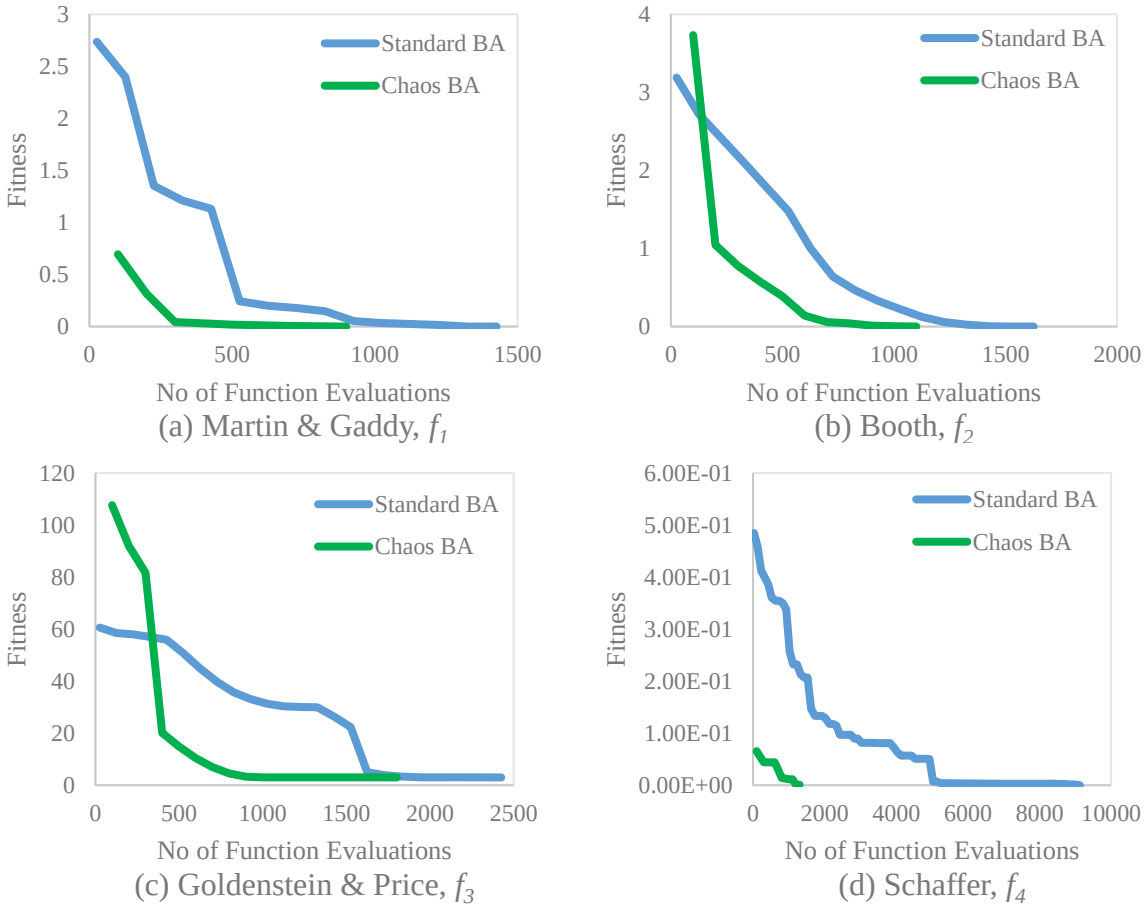


Figure 4.3: Sample graphs of convergence for standard Bees Algorithm and ChaosBA

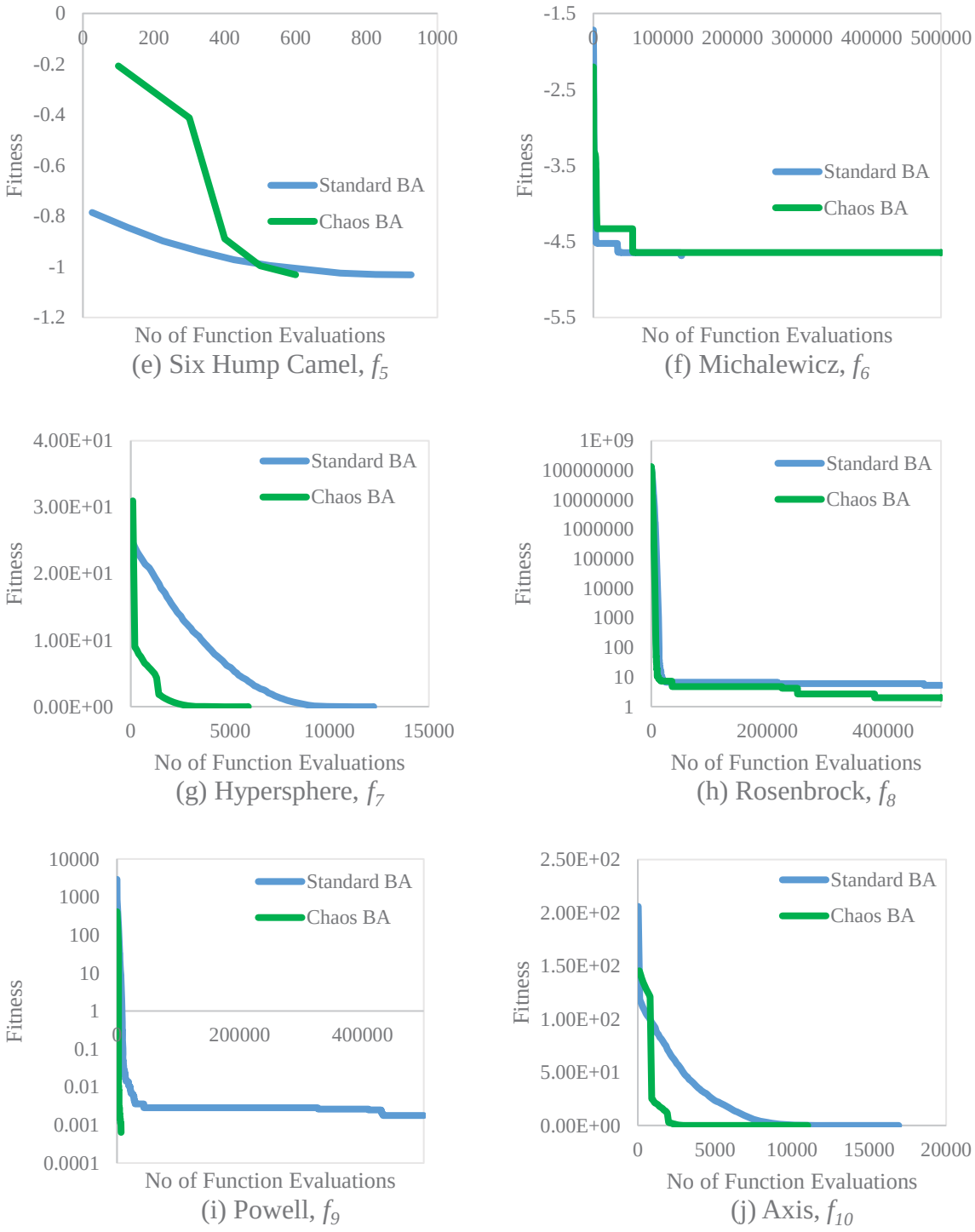


Figure 4.3: Sample graphs of convergence for standard Bees Algorithm and ChaosBA (continued)

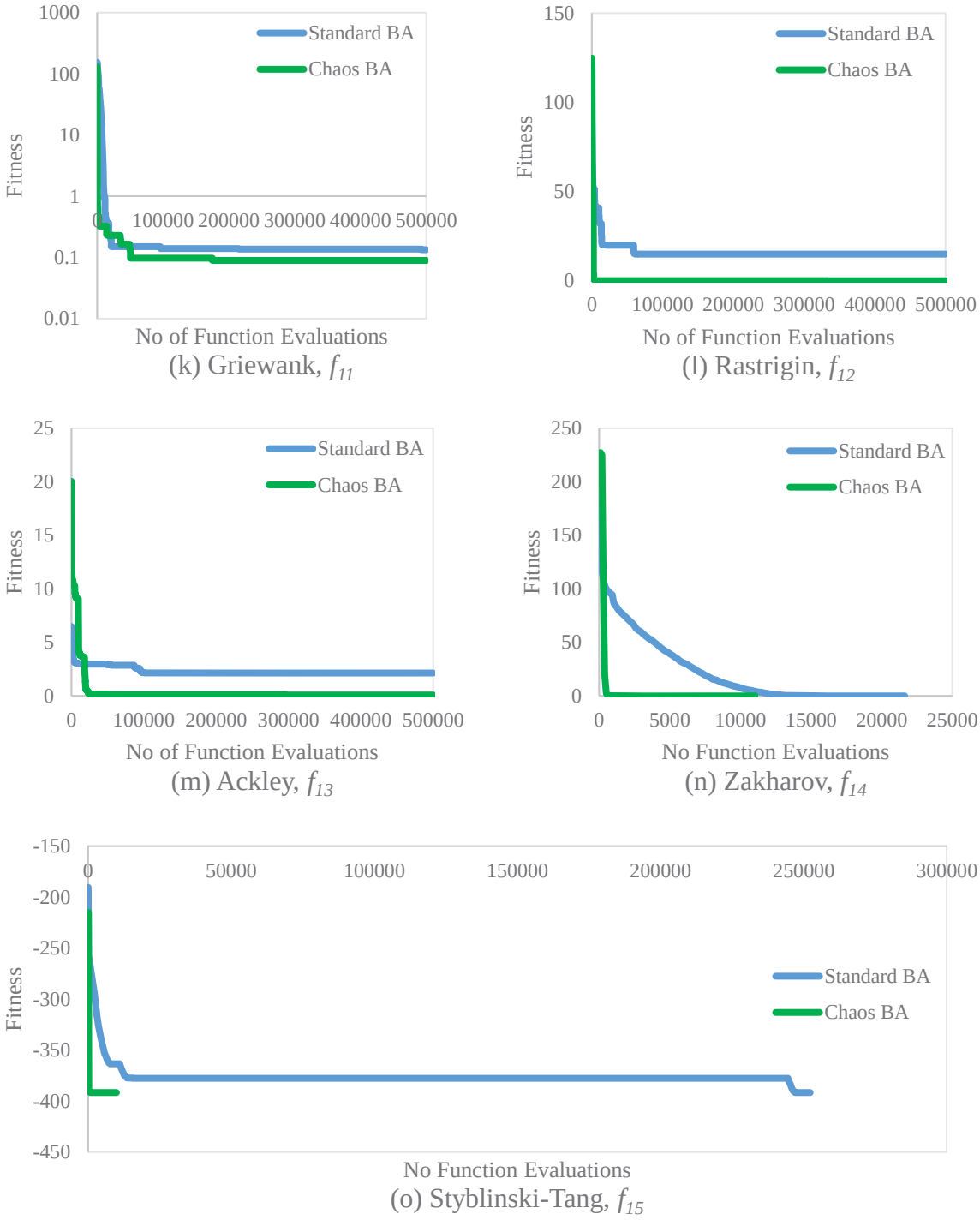


Figure 4.3: Sample graphs of convergence for standard Bees Algorithm and ChaosBA (continued)

In order to test the performance of the proposed method, SPSO2011 and qABC were introduced for comparison as in Chapter 3. The comparison of the algorithms performance in the fifteen benchmark functions is shown in Table 4.3 and the results of the significance test are presented in Table 4.4. The results that show significance over the other is written in boldface. Overall, ChaosBA outperformed SPSO2011 and qABC in eleven out of fifteen benchmark functions. The effectiveness of chaos in ChaosBA helped the algorithm to perform faster in unimodal functions like *Martin & Gaddy*, *Booth*, *Hypersphere*, *Powell* and *Axis*. However, ChaosBA could not surpass the performance of qABC in *Rosenbrock.* Additionally, the performance of ChaosBA was significantly better than SPSO2011 and qABC in multimodal functions such as *Goldenstein & Price, Schaffer, Six Hump Camel, Griewank, Zakharov* and *Styblinski-Tang*. Nonetheless, qABC showed better results than ChaosBA in *Michalewicz, Ackley* and *Rastrigin*.

As mentioned in the previous chapter, the algorithms were run 50 times for each benchmark function. To observe the consistency of the results, comparison graphs of the algorithm's performance in every benchmark function were exhibited in Figures 4.4 through 4.18. The speed of the algorithms was compared to assess if most of the algorithms found the optimum value of the function while the accuracy of the algorithms was compared to evaluate if most of the algorithms had reached the maximum function evaluations without converging to the global optimum. The blue box in some of the figures represents the zoomed area of the graph for clear comparison. As seen from the graphs, the ChaosBA performances are quite consistent in all 50 runs for each benchmark function. These graphs indicate that the implementation of the chaos method in the proposed algorithm not only improved the accuracy and speed but also helped to maintain the consistency of the algorithm in most of the benchmark functions.

Table 4.3: Performance comparison between ChaosBA, SPSO2011 and qABC

Function	ChaosBA			SPSO2011			qABC		
	Success	Accuracy	Speed	Success	Accuracy	Speed	Success	Accuracy	Speed
f_1	50	3.99E-04	**900**	50	5.42E-05	2800	38	8.82E-04	20223.5
f_2	50	4.21E-04	**1000**	50	0.00E+00	3200	48	3.90E-04	2050
f_3	50	4.27E-04	**1802.5**	50	5.00E-05	4000	50	2.16E-04	2450
f_4	50	1.63E-04	**2552**	50	0.00E+00	3500	32	7.55E-04	102706
f_5	50	2.83E-04	**650**	50	3.60E-05	3100	50	4.31E-04	950
f_6	23	4.12E-02	500000	25	3.20E-03	433650	50	**4.71E-04**	**111436.5**
f_7	50	8.11E-04	**5058**	50	8.67E-05	10750	50	6.78E-04	92470
f_8	0	2.75E+00	500000	4	1.08E+01	500000	0	**9.27E-02**	500000
f_9	50	7.74E-04	**38294**	50	9.96E-05	88450	36	9.83E-04	269100.5
f_{10}	50	8.29E-04	**10274**	50	8.90E-05	13500	50	6.08E-04	117207.5
f_{11}	0	7.26E-02	500000	0	3.18E-01	500000	18	**1.63E-03**	**500000**
f_{12}	0	**1.01E-01**	500000	6	3.56E-02	500000	0	6.27E-02	500000
f_{13}	0	1.06E-02	500000	0	5.93E+00	500000	43	**7.80E-04**	**372419.5**
f_{14}	50	**7.90E-04**	**11881**	50	1.00E-04	44950	0	4.06E-02	500000
f_{15}	50	6.03E-04	**10272**	6	2.83E+01	500000	41	6.51E-04	144582

Table 4.4: Statistical comparison between ChaosBA, SPSO2011 and qABC

Function	ChaosBA			
	SPSO2011		qABC	
	Accuracy	Speed	Accuracy	Speed
f_1	1.0000	**0.0000**	1.0000	**0.0000**
f_2	1.0000	**0.0000**	1.0000	**0.0000**
f_3	1.0000	**0.0000**	1.0000	**0.0198**
f_4	1.0000	**0.0332**	1.0000	**0.0002**
f_5	1.0000	**0.0000**	1.0000	**0.0000**
f_6	0.4777	0.0536	**0.0000**	**0.0000**
f_7	1.0000	**0.0000**	1.0000	**0.0000**
f_8	**0.0000**	0.4902	**0.0000**	1.0000
f_9	1.0000	**0.0000**	1.0000	**0.0000**
f_{10}	1.0000	**0.0000**	1.0000	**0.0000**
f_{11}	**0.0000**	1.0000	**0.0000**	**0.0000**
f_{12}	**0.0000**	0.3030	**0.0000**	1.0000
f_{13}	**0.0000**	1.0000	**0.0000**	**0.0000**
f_{14}	1.0000	**0.0000**	**0.0000**	**0.0000**
f_{15}	**0.0000**	**0.0000**	0.6599	**0.0000**

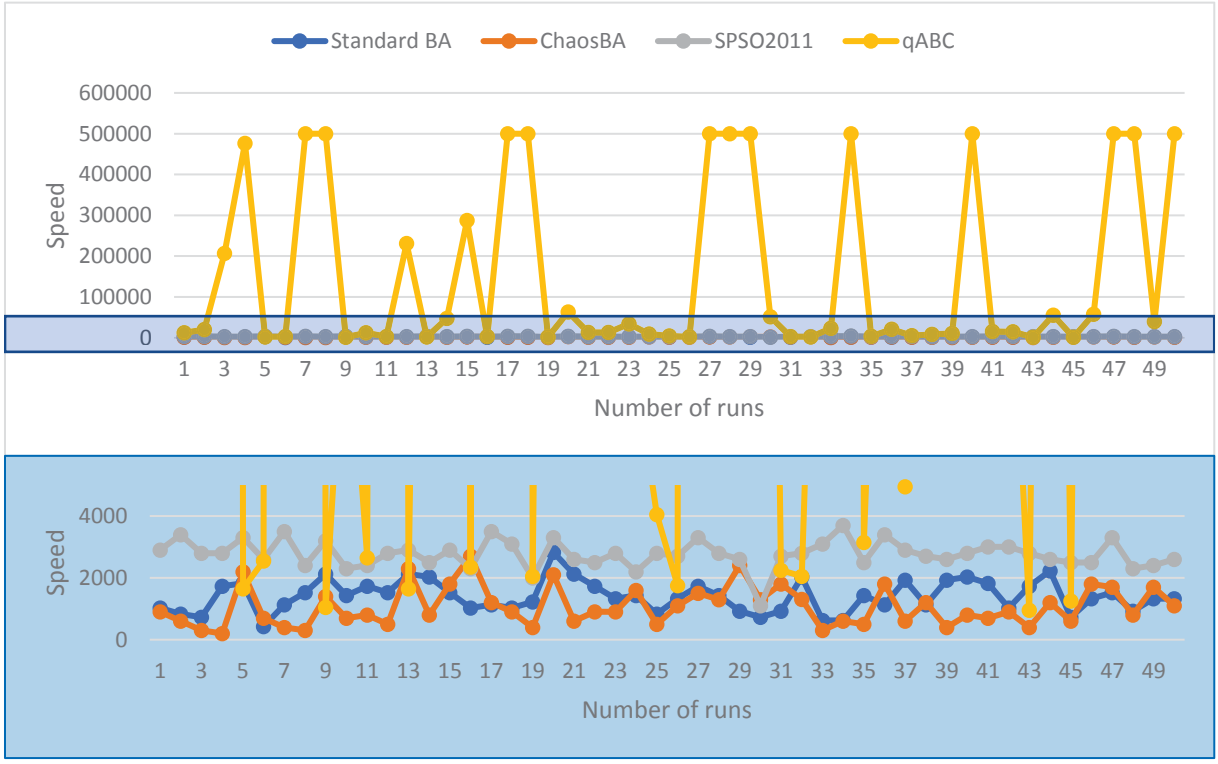


Figure 4.4: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_1

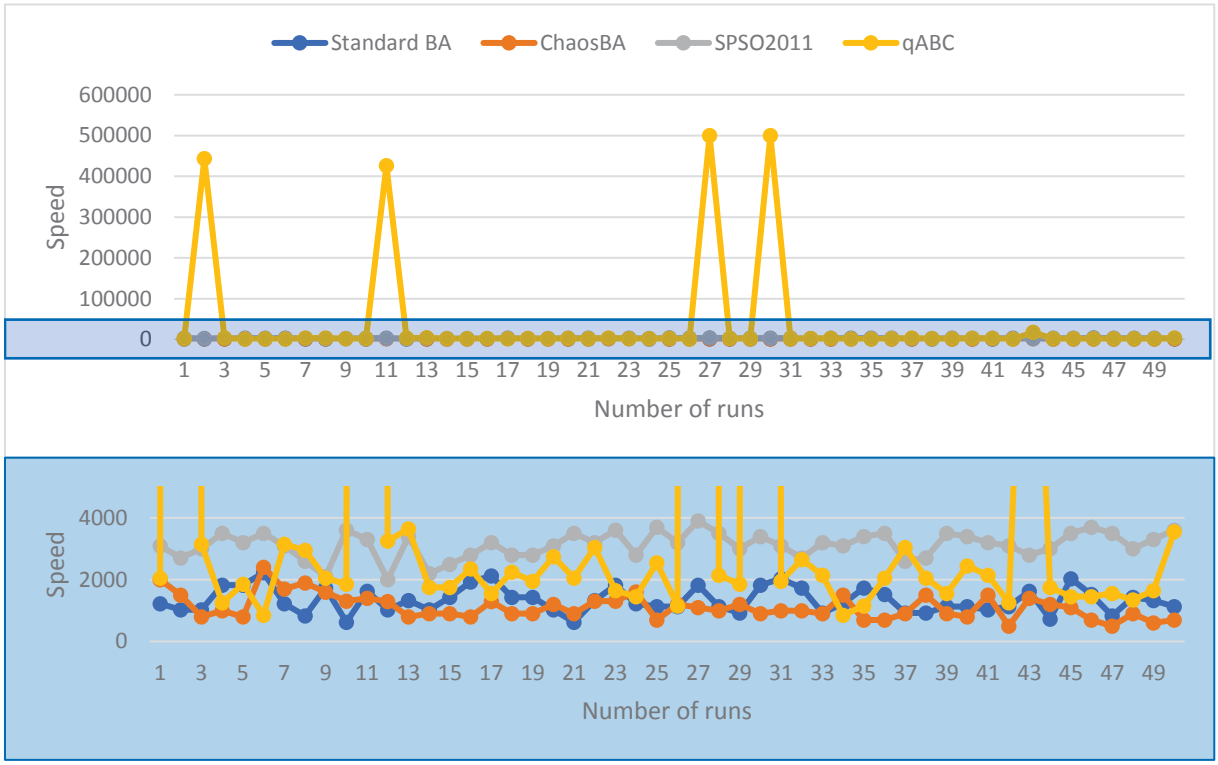


Figure 4.5: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_2

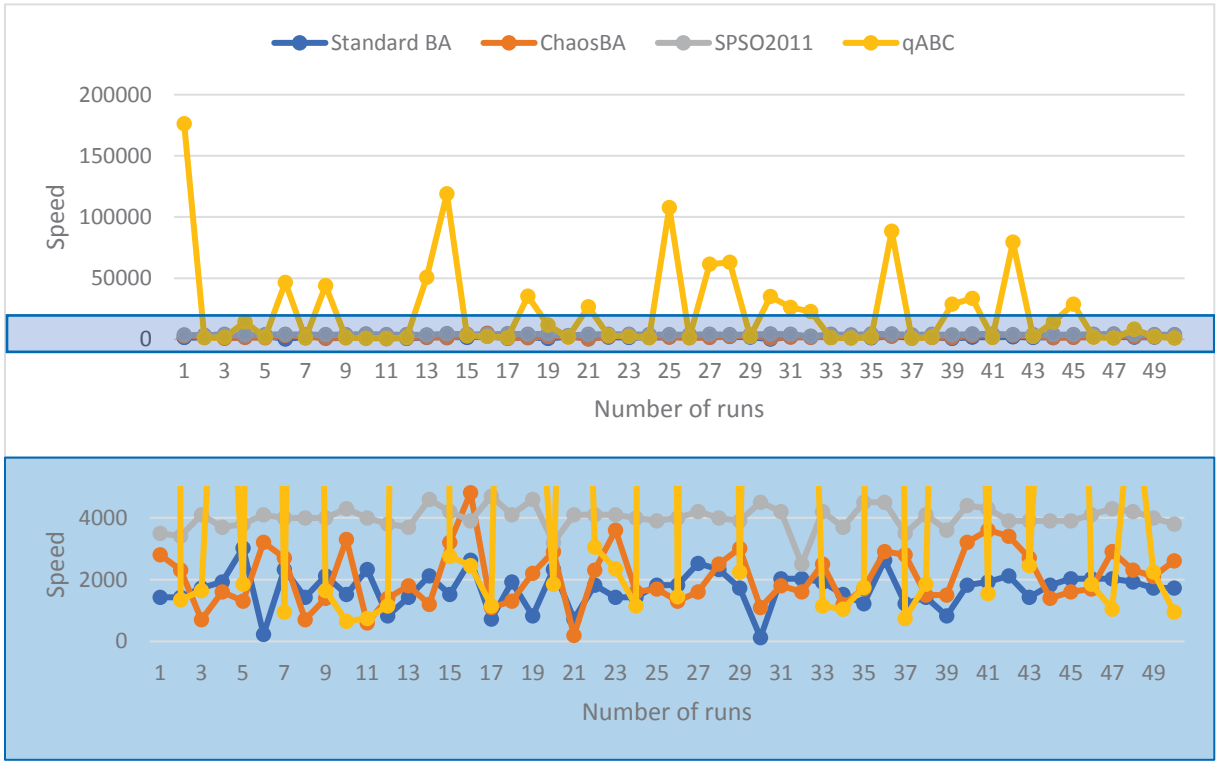


Figure 4.6: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_3

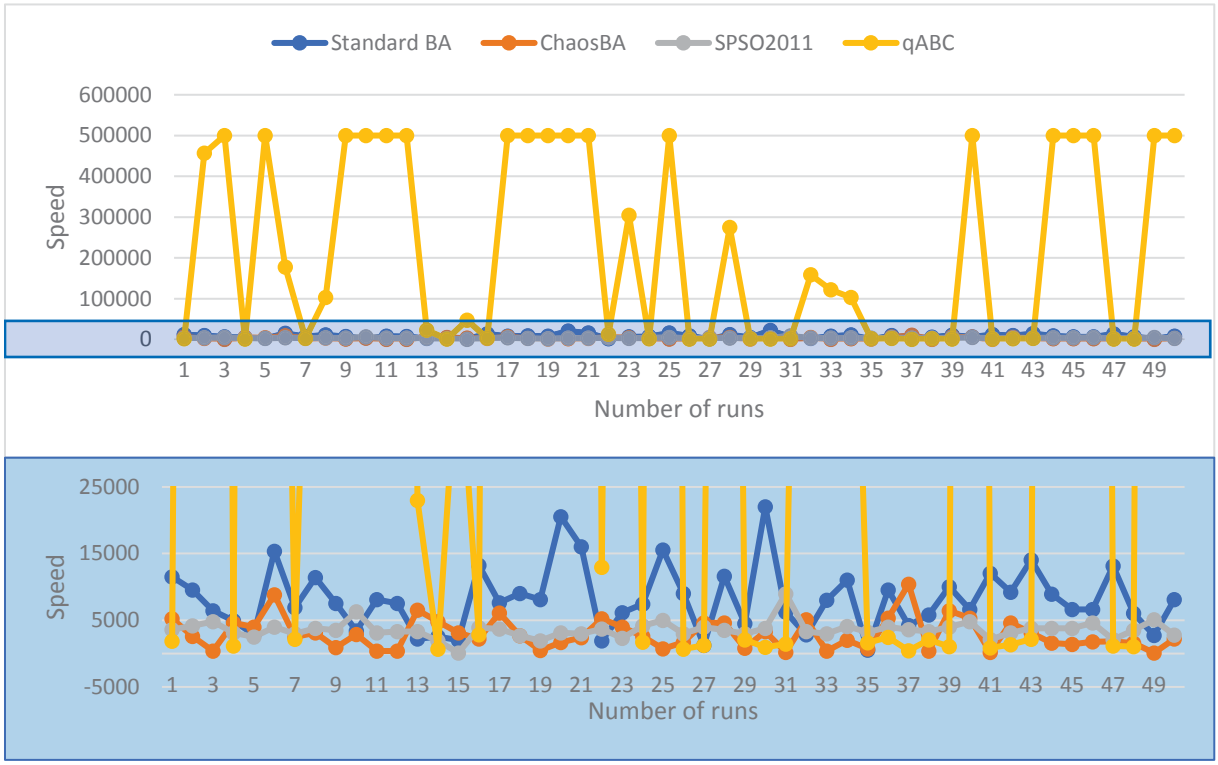


Figure 4.7: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_4

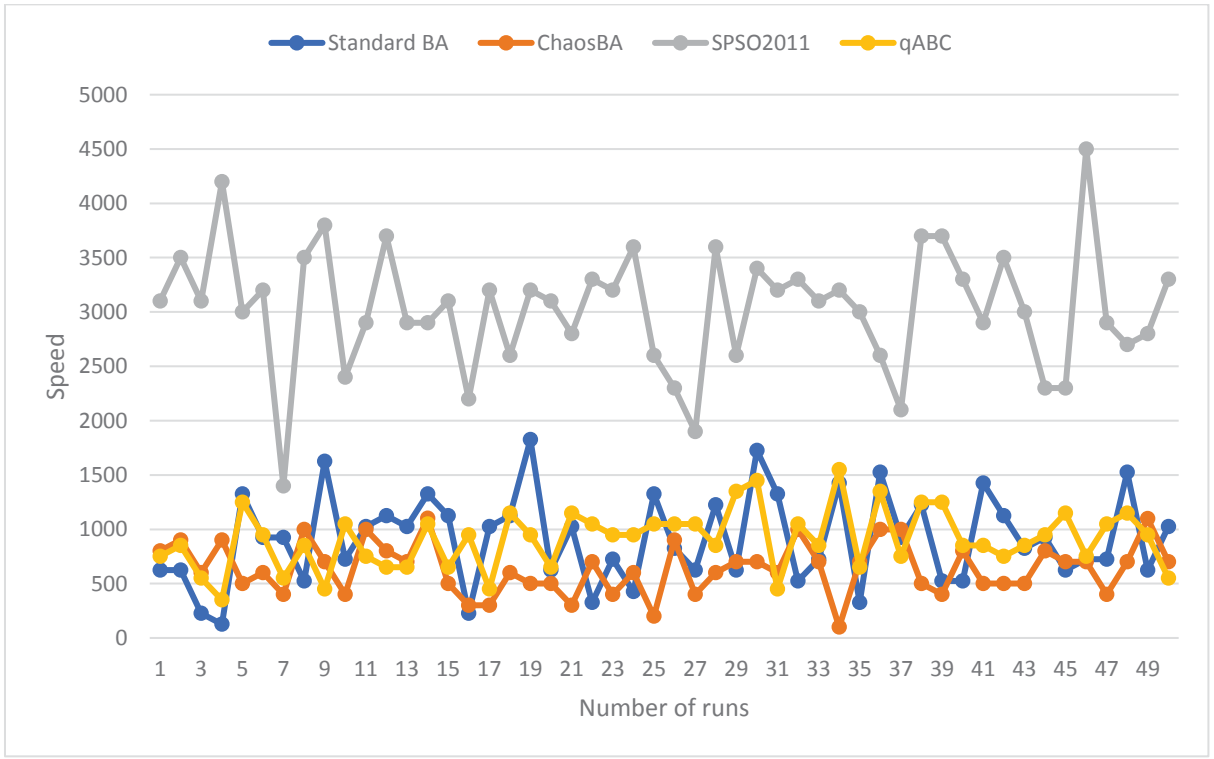


Figure 4.8: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_5

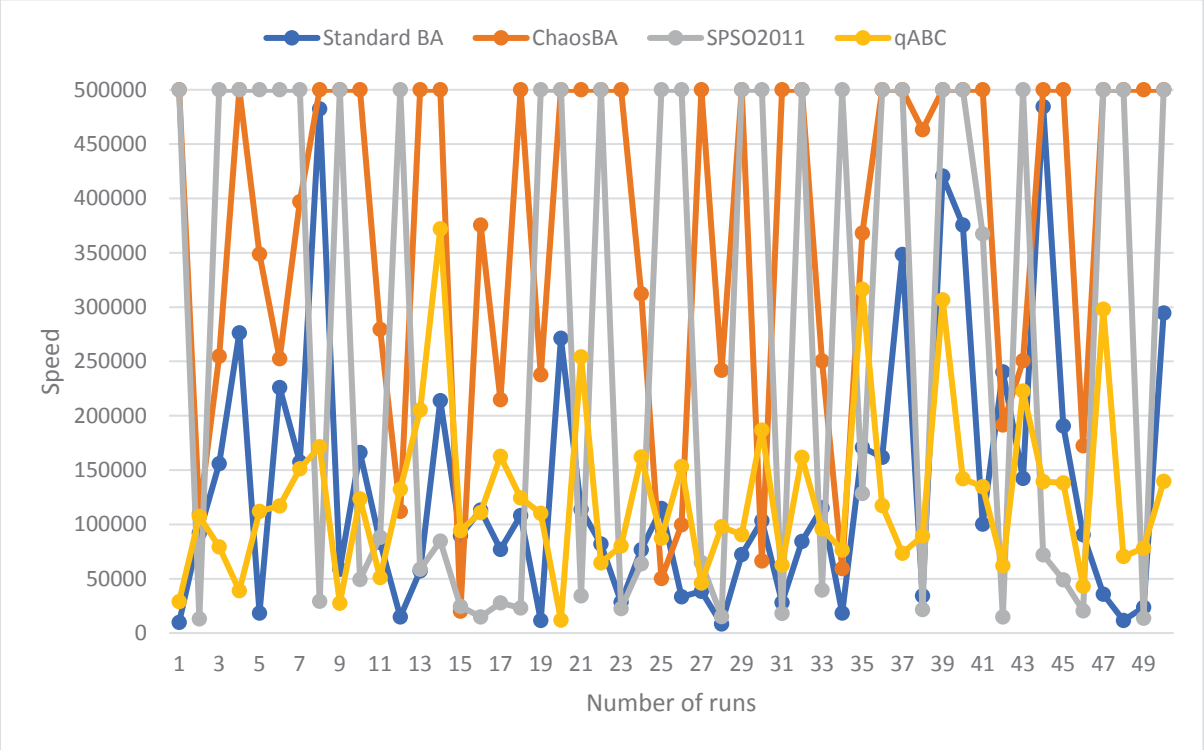


Figure 4.9: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_6

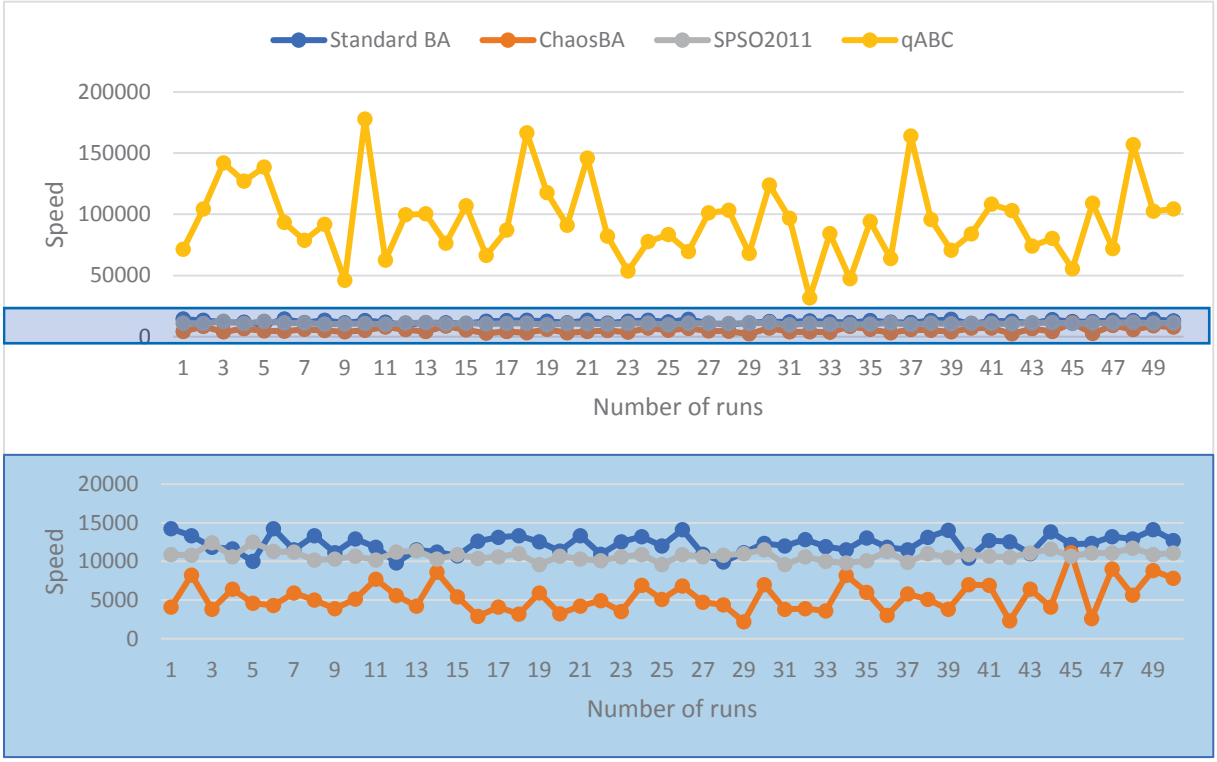


Figure 4.10: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_7

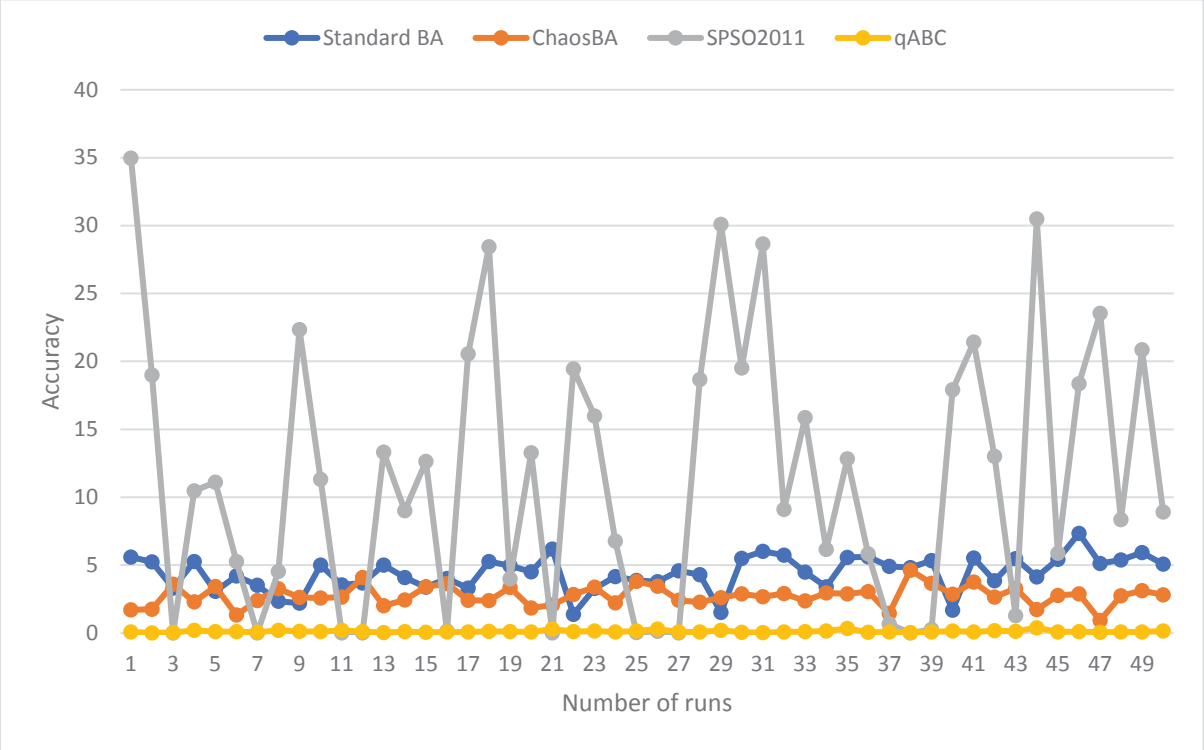


Figure 4.11: Comparison graph of accuracy performance between ChaosBA, SPSO2011 and qABC for f_8

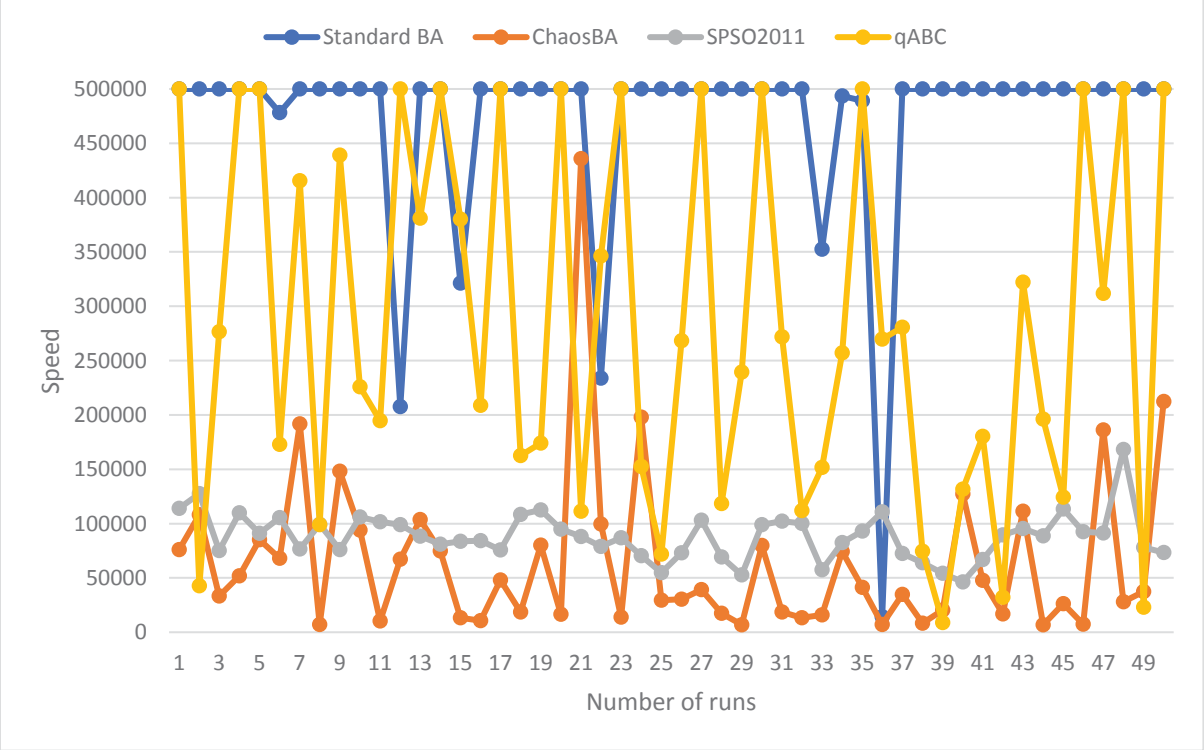


Figure 4.12: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_9

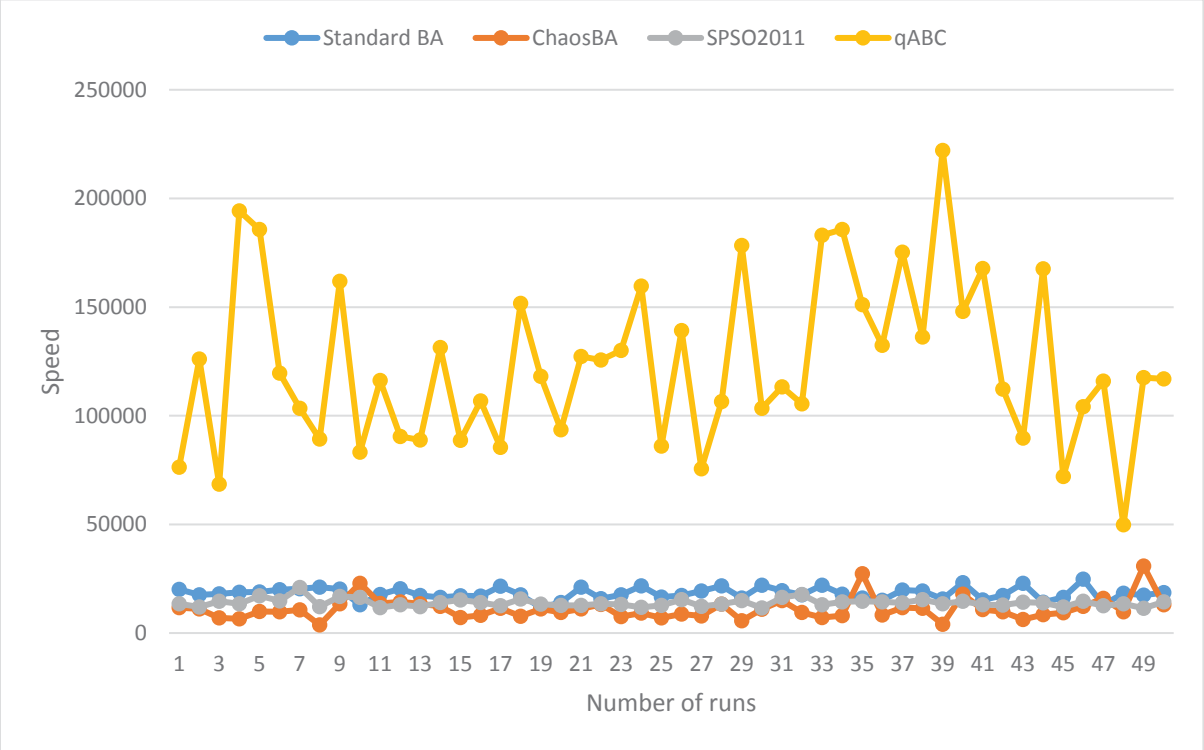


Figure 4.13: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_{10}

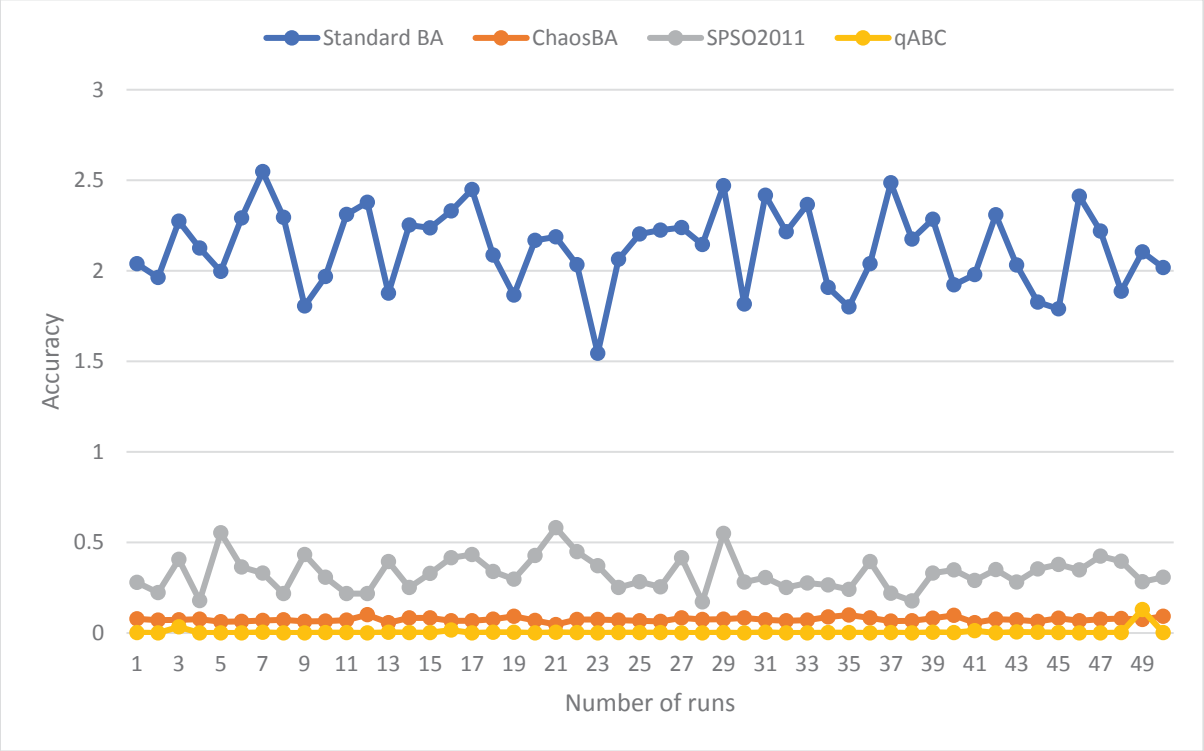


Figure 4.14: Comparison graph of accuracy performance between ChaosBA, SPSO2011 and qABC for f_{11}

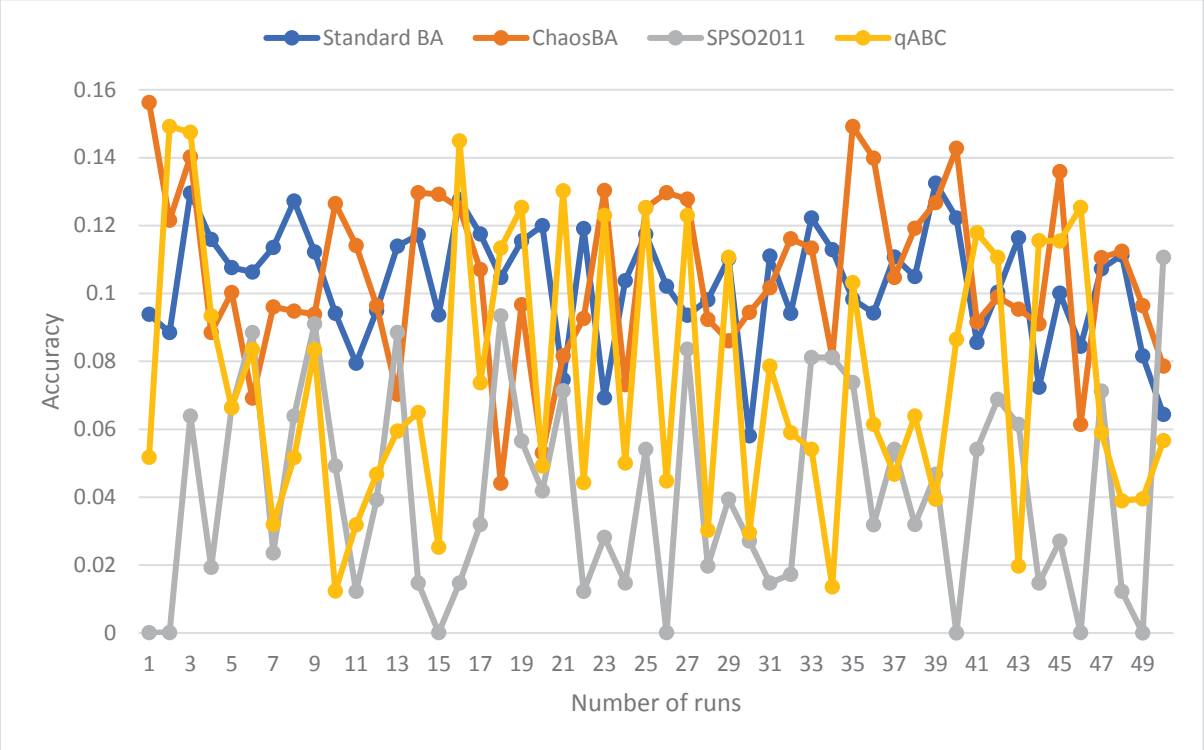


Figure 4.15: Comparison graph of accuracy performance between ChaosBA, SPSO2011 and qABC for f_{12}

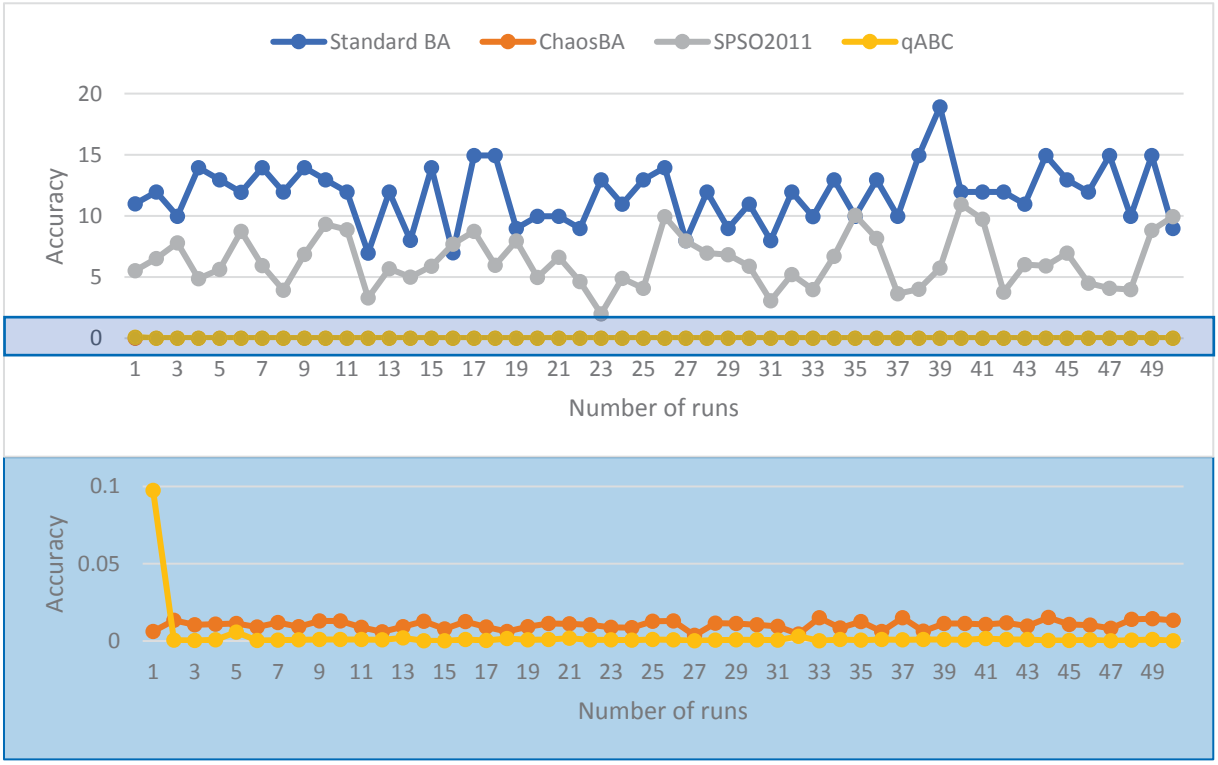


Figure 4.16: Comparison graph of accuracy performance between ChaosBA, SPSO2011 and qABC for f_{13}

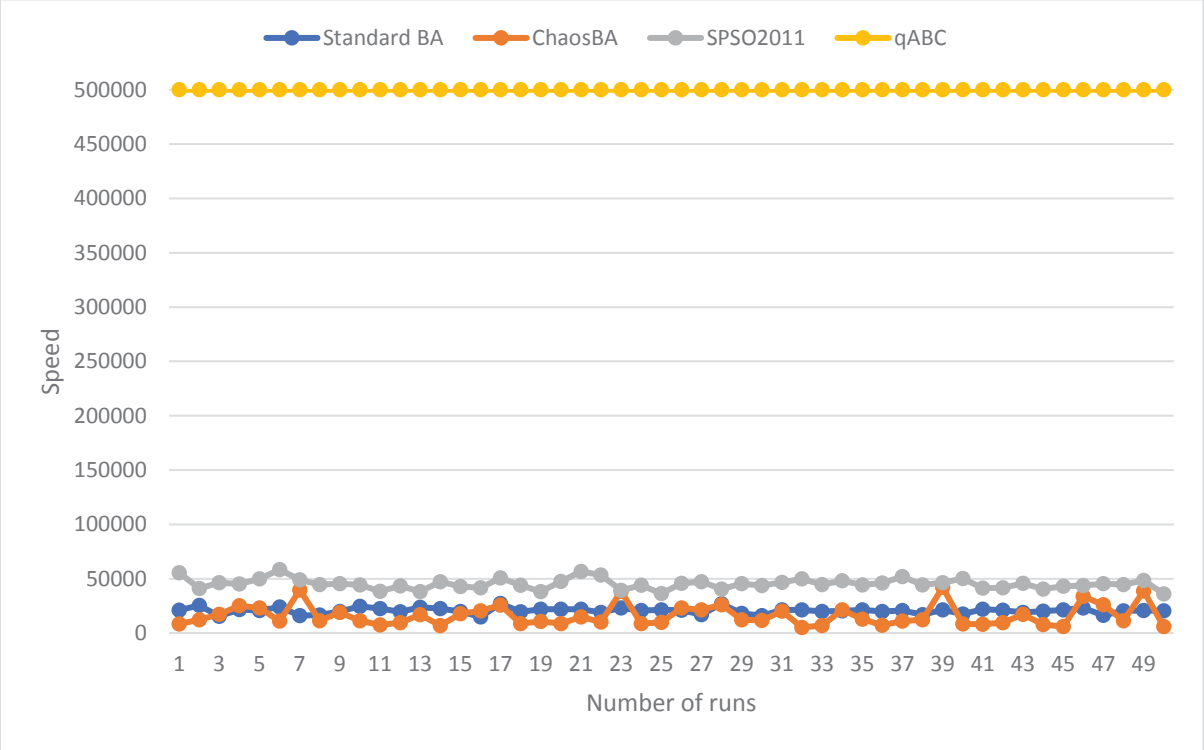


Figure 4.17: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_{14}

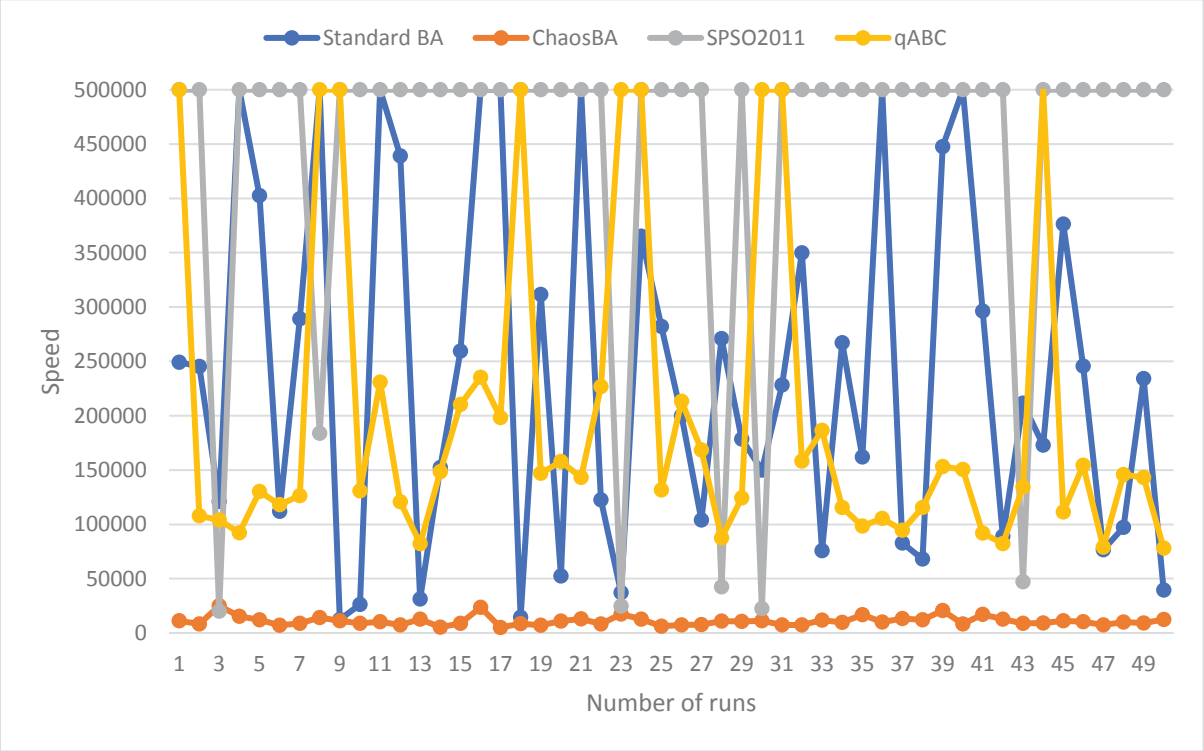


Figure 4.18: Comparison graph of speed performance between ChaosBA, SPSO2011 and qABC for f_{15}

4.5 Engineering benchmark constrained and mechanical design problems

To further test the proposed optimisation method's capability of handling more complex real problems, the proposed ChaosBA was tested on a set of constrained mechanical design problem with experimental setup as explained in Section 3.5. To make a fair comparison throughout the experiment, a similar parameter setting as in Table 3.7 was adopted in this exercise. Correspondingly, ChaosBA was tested 30 times under the same environment using a computer with an Intel Xeon 2.40 GHz processor and 8GB of RAM. The maximum number of function evaluations was set to 30,000. The results of the ChaosBA were compared to the standard BA and other well-known algorithms from the literature. The best solutions and parameters obtained by ChaosBA for the benchmark mechanical design problems are presented in Table 4.5. Further, Table 4.6 presents the statistical comparison between ChaosBA and the standard version of BA. Similarly, the Mann-Whitney significance test was used in the comparison and the statistically significant median and the best solution are shown in boldface.

As seen in Table 4.6, ChaosBA produces the best solutions in all five constrained benchmark mechanical design problems tested in this exercise. In the median solutions of the 30 independent runs for each problem, ChaosBA results were significantly better than the standard BA in the speed reducer problem. However, the experimental results reveal that the performance of ChaosBA in the three-truss bar, spring and welded beam were comparable to the standard BA based on the test by Mann-Whitney that show non-significant values for the aforementioned problems. On the contrary, the standard BA showed a significantly better performance than ChaosBA in the pressure vessel problem. The implementation of chaos in ChaosBA seemed to hamper the performance of the algorithm in this design problem.

Table 4.5: Best results obtained by ChaosBA for constrained benchmark mechanical design problems

	Three-truss bar	**Tension/compression spring**	**Pressure vessel**	**Welded beam**	**Speed reducer**
$f(x)$	263.8959558	0.012668692	6102.338976	1.735254909	2999.052075
x_1	0.788626505	0.052012134	0.8125	0.20456954	3.500016455
x_2	0.40838696	0.364513649	0.4375	3.502195764	0.700000015
x_3		10.84720497	41.99251914	9.084976954	17
x_4			178.8954069	0.205670748	7.449262502
x_5					7.847933958
x_6					3.351446569
x_7					5.2867058

Table 4.6: Comparison of the statistical results obtained from ChaosBA and standard BA for constrained benchmark mechanical design problems

Problem		**Standard Bees Algorithm**	**ChaosBA**
Three-bar truss	Best	263.8964263	**263.8959558**
	Median	263.9015189	263.9037329
	Mean	263.9032913	263.9048236
	SD	0.005504271	0.00682598
	Worst	263.919459	263.9226124
	Evaluations	30,000	30,000
Tension/compression spring	Best	0.012670733	**0.012668692**
	Median	0.012681849	0.012698967
	Mean	0.012760301	0.012798607
	SD	0.00017804	0.000245459
	Worst	0.013402018	0.013712851
	Evaluations	30,000	30,000
Pressure vessel	Best	6119.246703	**6102.338976**
	Median	**6222.419644**	6654.207513
	Mean	6259.109338	6590.201192
	SD	126.2859774	234.6677032
	Worst	6522.111587	7050.021657
	Evaluations	30,000	30,000
Welded beam	Best	1.73958454	**1.735254909**
	Median	1.767187634	1.758684186
	Mean	1.766361958	1.762479433
	SD	0.0150332	0.014575534
	Worst	1.810428812	1.793574405
	Evaluations	30,000	30,000
Speed reducer	Best	3006.240405	**2999.052075**
	Median	3038.251131	**3027.828475**
	Mean	3035.608189	3025.771844
	SD	10.27826125	10.98674355
	Worst	3050.226338	3051.562288
	Evaluations	30,000	30,000

Further comparison was made to the results of ChaosBA and other algorithms in the engineering application design problems as in the previous chapter. In this case, SCA and PSO-DE were compared in the three-bar truss problem, while GA1, GA2, PSO-DE, UPSOm, and ABC were used in the spring, pressure vessel, and welded beam problems. Also, comparison was made with SC, PSO-DE and ABC in the speed reducer problem. Tables 4.7 to 4.11 exhibit the comparison results for all constrained mechanical design problems. In all five design problems, the results show that the ChaosBA performance is comparable to other algorithms as far as the best solutions are concerned.

Table 4.7: ChaosBA comparison results for the three-truss bar optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
SC	263.895846	263.903356	263.969756	17,610
PSO-DE	263.89584338	263.89584338	263.89584338	17,600
Bees Algorithm	263.8964263	263.9032913	263.919459	30,000
ChaosBA	263.8959558	263.9048236	263.9226124	30,000

Table 4.8: ChaosBA comparison results for the tension/compression spring optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
GA3	0.0127048	0.0127690	0.0128220	900,000
GA4	0.0126810	0.0127420	0.0129730	80,000
PSO-DE	0.012665233	0.012665233	0.012665233	42,100
UPSOm	0.0131200	0.0229478	-	100,000
ABC	0.012665	0.012709	-	30,000
Bees Algorithm	0.012670733	0.012760301	0.013402018	30,000
ChaosBA	0.012668692	0.012798607	0.013712851	30,000

Table 4.9: ChaosBA comparison results for the pressure vessel optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
GA3	6288.7445	6293.8432	6308.4970	900,000
GA4	6059.9463	6177.2533	6469.3220	80,000
PSO-DE	6059.714335	6059.714335	6059.714335	42,100
UPSOm	6544.27	9032.55	-	100,000
ABC	6059.714736	6245.308144	-	30,000
Bees Algorithm	6119.246703	6259.109338	6522.111587	30,000
ChaosBA	6102.338976	6590.201192	7050.021657	30,000

Table 4.10: ChaosBA comparison results for the welded beam optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
GA3	1.748309	1.771973	1.785835	900,000
GA4	1.728226	1.792654	1.993408	80,000
PSO-DE	1.724852309	1.724852309	1.724852309	66,600
UPSOm	1.92199	2.83721	-	100,000
ABC	1.724852	1.741913	-	30,000
Bees Algorithm	1.73958454	1.766361958	1.810428812	30,000
ChaosBA	1.735254909	1.762479433	1.793574405	30,000

Table 4.11: ChaosBA comparison results for the speed reducer optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
SC	2994.744241	3001.758264	3009.964736	54,456
PSO-DE	2996.348165	2996.348165	2996.348166	70,100
ABC	2997.058412	2997.058412	-	30,000
Bees Algorithm	3006.240405	3035.608189	3050.226338	30,000
ChaosBA	2999.052075	3025.771844	3051.562288	30,000

4.6 Summary

This chapter presented the Bees Algorithm with chaos (ChaosBA) which intended to improve the convergence speed and solution accuracy of the standard BA. A well-known logistic map was utilised to generate the chaotic sequences for local and global search procedures in the proposed algorithm. Randomly generated points in the standard BA were replaced by these chaotic sequences in the deployment method of the recruited bees. The effectiveness of the ChaosBA was evaluated in fifteen unconstrained benchmark functions with different dimensions and characteristics, as well as five constrained mechanical design problems.

The proposed algorithm performed effectively for the unconstrained benchmark functions compared to the standard BA. The experimental results revealed that the proposed algorithm is an efficient and effective algorithm in comparison with the standard BA including difficult

multimodal functions such as *Rosenbrock*, *Powell*, *Ackley*, and *Rastrigin*. Comparison results with another two state-of-the-art algorithms also provided evidence that the ChaosBA performance is significantly better for most of the test functions. The efficiency of the proposed algorithm was due to its impressive combination of pseudo-randomness, ergodicity, and irregularity properties of chaotic approach of the local search in avoiding being trapped at local optima and the desirable global search ability of the ChaosBA. The results proved that a chaotic sequence can have a noticeable effect on the performance of ChaosBA for unconstrained benchmark functions. Additionally, ChaosBA found the best solution in all constrained design problems and performs effectively for engineering applications such as in the speed reducer problem when compared to the standard BA. Most importantly, the performance of ChaosBA in the design problems was found to be comparable with other well-known algorithms in the literature.

CHAPTER 5

BEES ALGORITHM WITH ESTIMATION DISTRIBUTION (BAED)

5.1 Preliminaries

The ability to use any information during the initial search to aid an algorithm in the next search phase is ideal for finding the optimum solution. Information from the current best solutions could provide the algorithm with a clue to which direction the next search should be conducted. Therefore, this chapter proposes a new variant of Bees Algorithm that utilises the Estimation Distribution Algorithm (EDA) method in the best solution found so far.

The remainder of this chapter is organised as follows. Section 5.2 presents the EDA method and the proposed Bees Algorithm with Estimation Distribution (BAED), Section 5.3 explains the experimental setup for unconstrained benchmark functions followed by the results and discussion in Section 5.4. The experimental setup and results for constrained mechanical engineering benchmark problems are presented and discussed in Section 5.5. Section 5.6 summarises the chapter.

5.2 Bees Algorithm with Estimation Distribution (BAED)

In this section, a new variant of Bees Algorithm is developed with the aim of using the current information of the best solutions to guide the search towards a promising area by sampling new solutions from a probability model inspired by EDA. BAED uses the EDA method on the “best

bees" before the local and global search procedures intended to provide solutions which are more promising than solutions currently generated.

EDA is a stochastic optimisation technique that searches for potential solutions in the search space by building and sampling probabilistic models to generate promising candidate solutions (Hauschild & Pelikan, 2011; Larrañaga & Lozano, 2002). In general, EDA typically works in three phases as follows (Ahn, An, & Yoo, 2012):

Step 1: Select good individuals from a population.

Step 2: Estimate the probability distribution from the selected individuals.

Step 3: Generate new individuals from the estimated distribution.

The probability distribution of the promising solutions can be modelled by the Gaussian distribution (Larrañaga & Lozano, 2002), Gaussian mixture (Bosman & Thierens, 2000) or a histogram (Tsutsui et al., 2001).

Initially, EDA generates a set of random individuals from the search space. These individuals are scored using the fitness function to give a numerical ranking on how accurate each individual is for the given problem. The higher the rank the better the solution. Based on this ranked population, a subset of the most promising solutions is selected by a truncation selection method. Then, EDA constructs a probabilistic model to estimate the probability distribution of the selected individuals. Once the model is constructed, a new set of individuals are generated by sampling the distribution. These newly generated individuals are then combined with the initial population. This process is iterated until the optimum solution is found or the number of iterations has reached a certain threshold.

The proposed BAED uses a similar method in EDA to generate new candidate solutions from the existing population during the initialisation phase of the standard BA. However, few modifications have been implemented in terms of the selection procedure, sampling of the probability distribution and number of candidate solutions to be generated. These modifications are important to suit this method into the learning mechanisms and parameters of BAED. Figure 5.1 illustrates the process of new population generation in BAED.

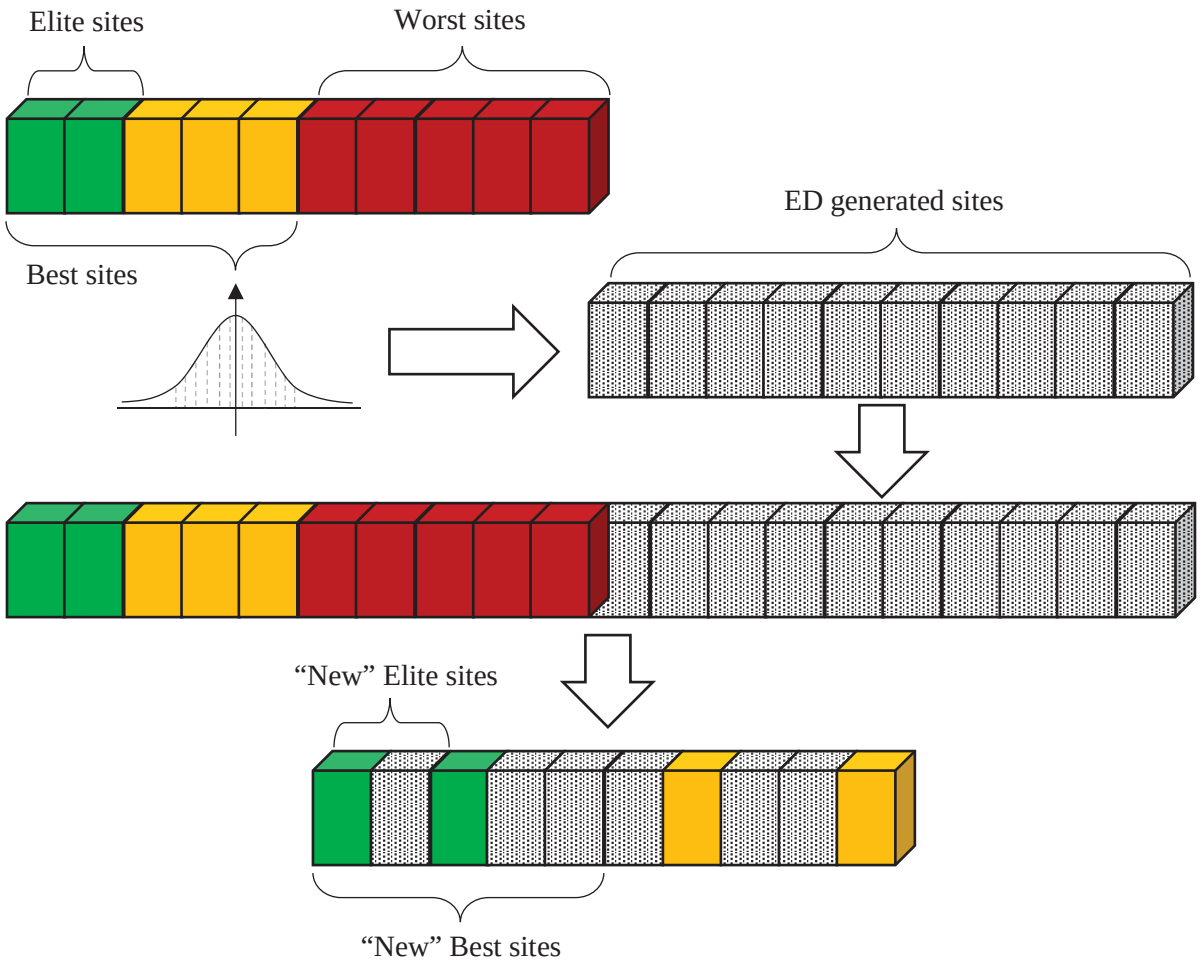


Figure 5.1: Population operation diagram for BAED

After the initialisation phase, the population is evaluated and ranked. From this ranked population, a subset of the most promising solutions is selected as *best sites* according to standard BA parameter. An example selection is *best sites nb* = 5 and *elite sites ne* = 2 as

demonstrated in Figure 5.1. Then, the algorithm estimates the probability distribution of the selected best sites and a probabilistic model using the Gaussian distribution is constructed. Thus, the new candidate solutions are generated by sampling the distribution induced by this model.

The candidate solutions are then evaluated and merged back into the old population to be ranked once again. Finally, the ranked population is truncated according to the parameter of BAED for *number of scout bees, ns.* The newly truncated population now has new *best sites* and *elite sites* to be carried over for the exploitation and exploration phases in local search and global search respectively. The process is repeated until some termination criteria are met. The termination criteria are typically when the optimum solution is found, or a number of iterations have elapsed. The overall procedure of BAED is outlined in Figure 5.2.

5.3 Experimental setup

In this chapter, the performance of BAED was tested on the same set of benchmark functions as in Section 3.3 and Section 4.3 (see Appendix A). Similar stopping criteria and parameters setting as in previous experimental setups in Chapter 3 and Chapter 4 was also used in this chapter for BAED as well as for SPSO2011 and qABC. For comparison purposes, the Mann-Whitney statistical significance test was used as in the previous chapters.

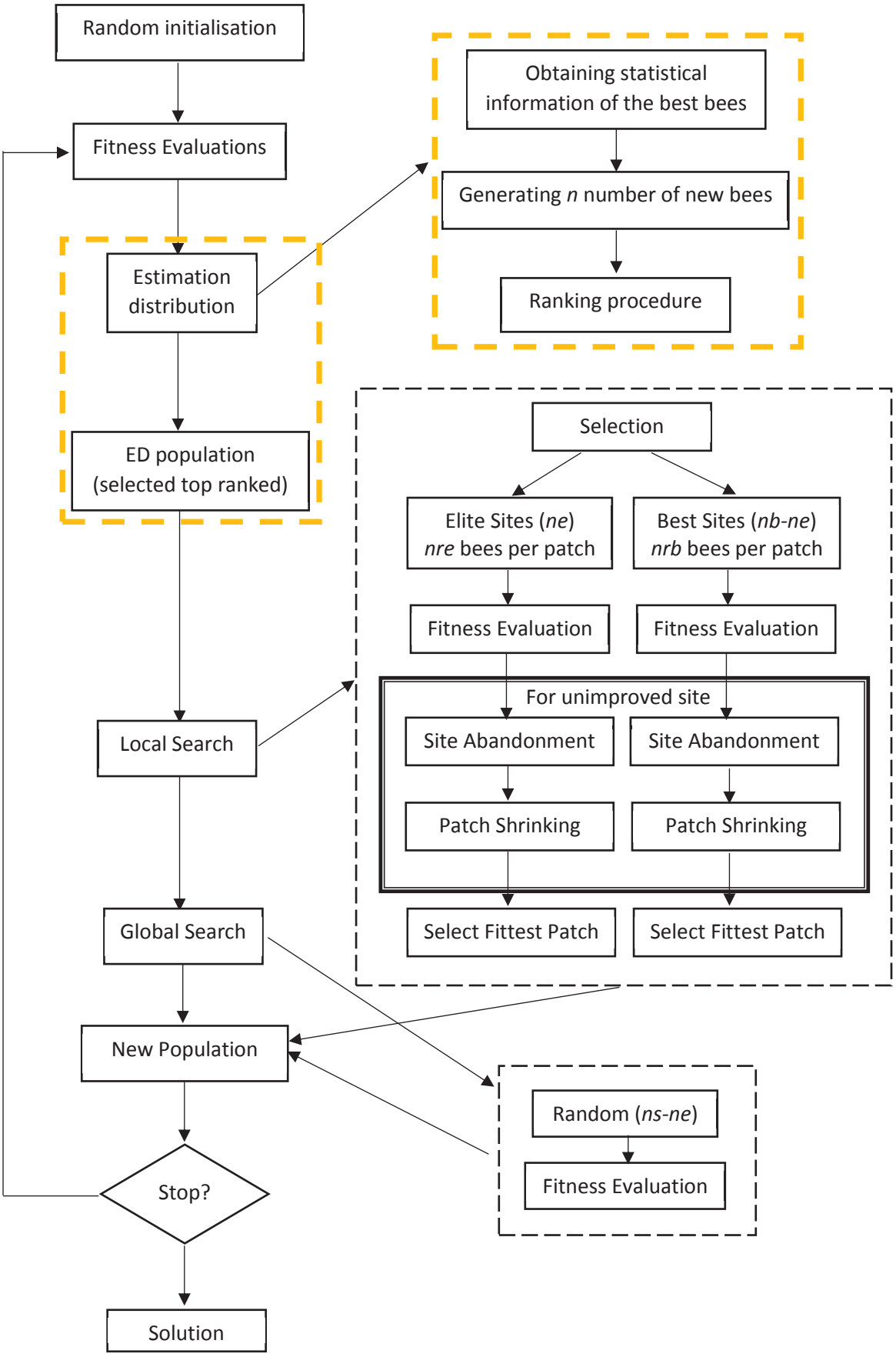


Figure 5.2: Flow chart of Bees Algorithm with estimation distribution (BAED)

5.4 Results and Discussion

Table 5.1 shows the comparison of BAED and the standard BA performances over 50 runs. Based on the median speed results in Table 5.1, the BAED demonstrates significantly better speed than the standard BA in all low dimensional benchmark functions (f_1-f_6). The comparison shows an improvement in speed which varies from 52% to 90%. In high dimensional benchmark functions (f_7-f_{15}), the BAED found an optimum value faster than standard BA in *Hypersphere, Powell, Axis, Zakharov* and *Styblinski-Tang.* Meanwhile, for functions *Ackley, Griewank,* and *Rastrigin* the BAED finds better median accuracy than the standard BA even though both algorithms failed to converge to the optimum values. Overall, the experimental results revealed that the proposed algorithm outperforms the standard BA in fourteen out of fifteen benchmark functions. Table 5.2 presents the summary of the statistical significance test conducted between the two algorithms. The algorithm that shows significance over the other is highlighted in boldface. Furthermore, Figures 5.2 to 5.7 show the convergence graphs of both algorithms for all fifteen benchmark functions.

Based on the performance of the proposed algorithm in the unconstrained benchmark functions simulation above, it is evident that the inclusion of the EDA method in utilising the current best population's information has helped the BAED to converge faster than the standard BA. The adoption of the EDA method in the local search procedures has increased the ability of the elite bees to find the most promising points in the search area by using the probabilistic model. Therefore, the efficiency of the proposed algorithm has been fully illustrated in the improvement of difficult functions for the standard BA including *Michalewicz, Powell, Axis, Zakharov* and *Styblinski-Tang*. However, the BAED performance in the *Rosenbrock* function

with the global optimum located inside a long, narrow, parabolic shaped flat valley has caused problems to the algorithm, thus failing to find the optimum value. The EDA method in the proposed algorithm was unable to help the local search in this type of function and the results showed that the standard BA accuracy was significantly better than the BAED even though both did not converge.

Table 5.1: Performance comparison between standard Bees Algorithm and BAED

Function	**Standard Bees Algorithm**			**BAED**		
	Success	**Accuracy**	**Speed**	**Success**	**Accuracy**	**Speed**
f_1	50	3.90E-04	1376	50	2.39E-04	**656**
f_2	50	3.81E-04	1226	50	2.58E-04	**530**
f_3	50	5.06E-04	1826	50	2.32E-04	**782**
f_4	50	2.39E-04	7583.5	50	1.10E-04	**782**
f_5	50	3.90E-04	926	50	2.03E-04	**404**
f_6	50	5.20E-04	96096.5	50	2.34E-04	**11748.5**
f_7	50	7.95E-04	12326	50	7.03E-04	**2231**
f_8	0	**4.54E+00**	500000	0	5.77E+00	500000
f_9	8	1.56E-03	500000	50	**9.34E-04**	**98939**
f_{10}	50	7.55E-04	17776	50	8.28E-04	**3176**
f_{11}	0	2.16E+00	500000	7	**1.32E-01**	500000
f_{12}	0	1.06E-01	500000	9	**1.51E-02**	500000
f_{13}	0	1.20E+01	500000	2	**1.99E+00**	500000
f_{14}	50	8.93E-04	20826	50	8.75E-04	**7964**
f_{15}	42	4.87E-04	231150	50	5.76E-04	**25104.5**

Table 5.2: Statistical comparison between standard Bees Algorithm and BAED

Function	**Standard Bees Algorithm**	
	BAED	
	Accuracy (p-value)	**Speed (p-value)**
f_1	1.0000	**0.0000**
f_2	1.0000	**0.0000**
f_3	1.0000	**0.0000**
f_4	1.0000	**0.0000**
f_5	1.0000	**0.0000**
f_6	1.0000	**0.0000**
f_7	1.0000	**0.0000**
f_8	**0.0111**	1.0000
f_9	**0.0000**	**0.0000**
f_{10}	1.0000	**0.0000**
f_{11}	**0.0000**	0.2301
f_{12}	**0.0000**	0.1211
f_{13}	**0.0000**	0.7279
f_{14}	1.0000	**0.0000**
f_{15}	0.8808	**0.0000**

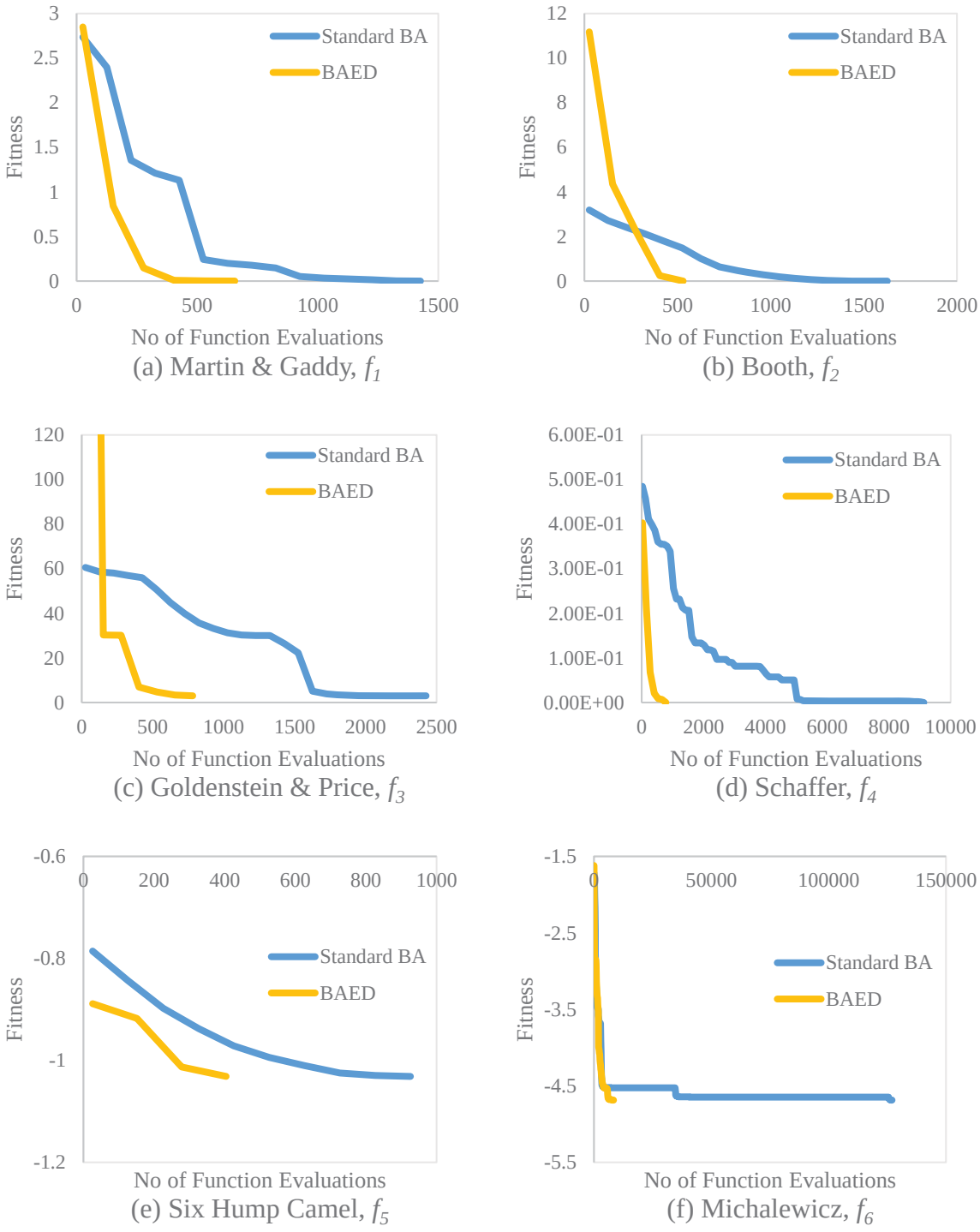
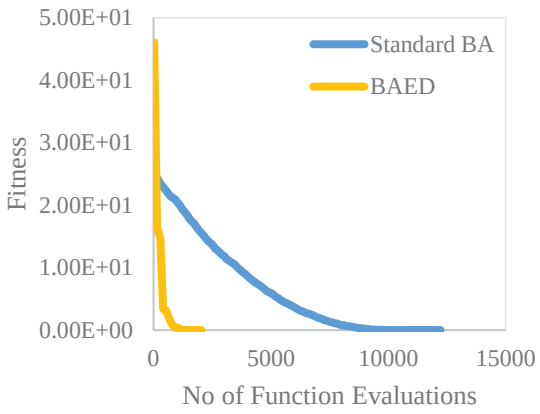
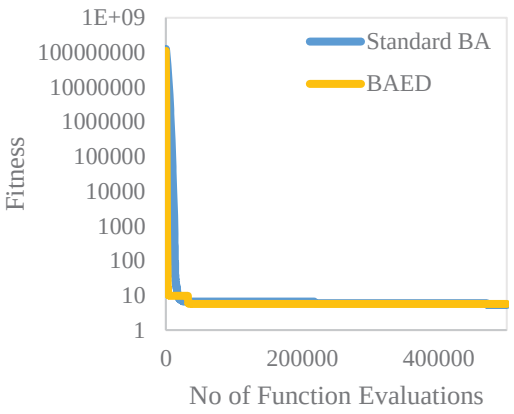


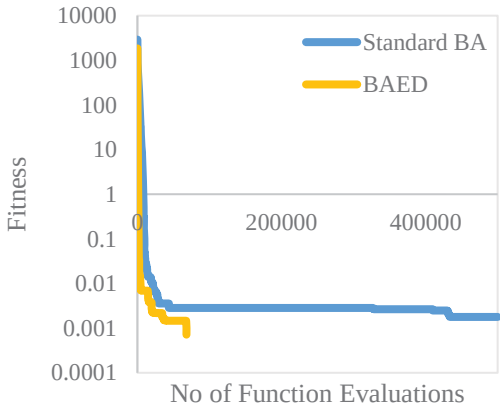
Figure 5.3: Sample graphs of convergence for standard Bees Algorithm and BAED



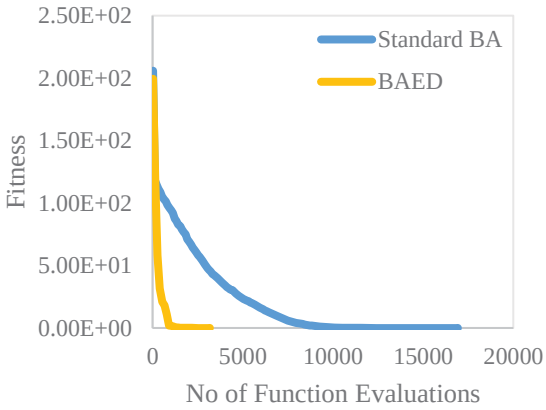
(g) Hypersphere, f_7



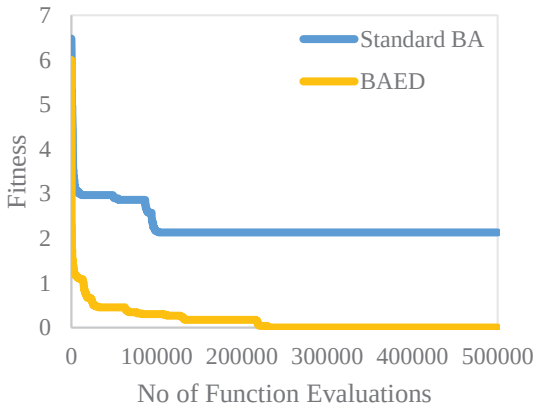
(h) Rosenbrock, f_8



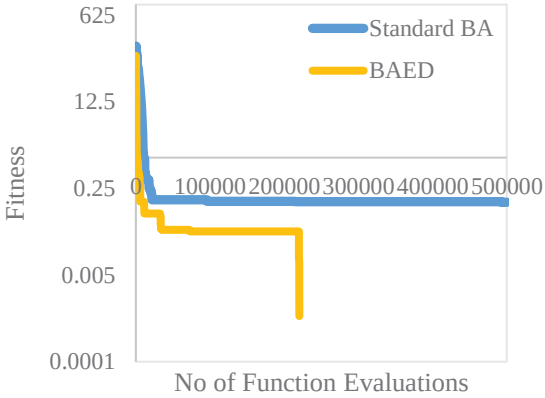
(i) Powell, f_9



(j) Axis, f_{10}



(k) Ackley, f_{11}



(l) Griewank, f_{12}

Figure 5.3: Sample graphs of convergence for standard Bees Algorithm and BAED (continued)

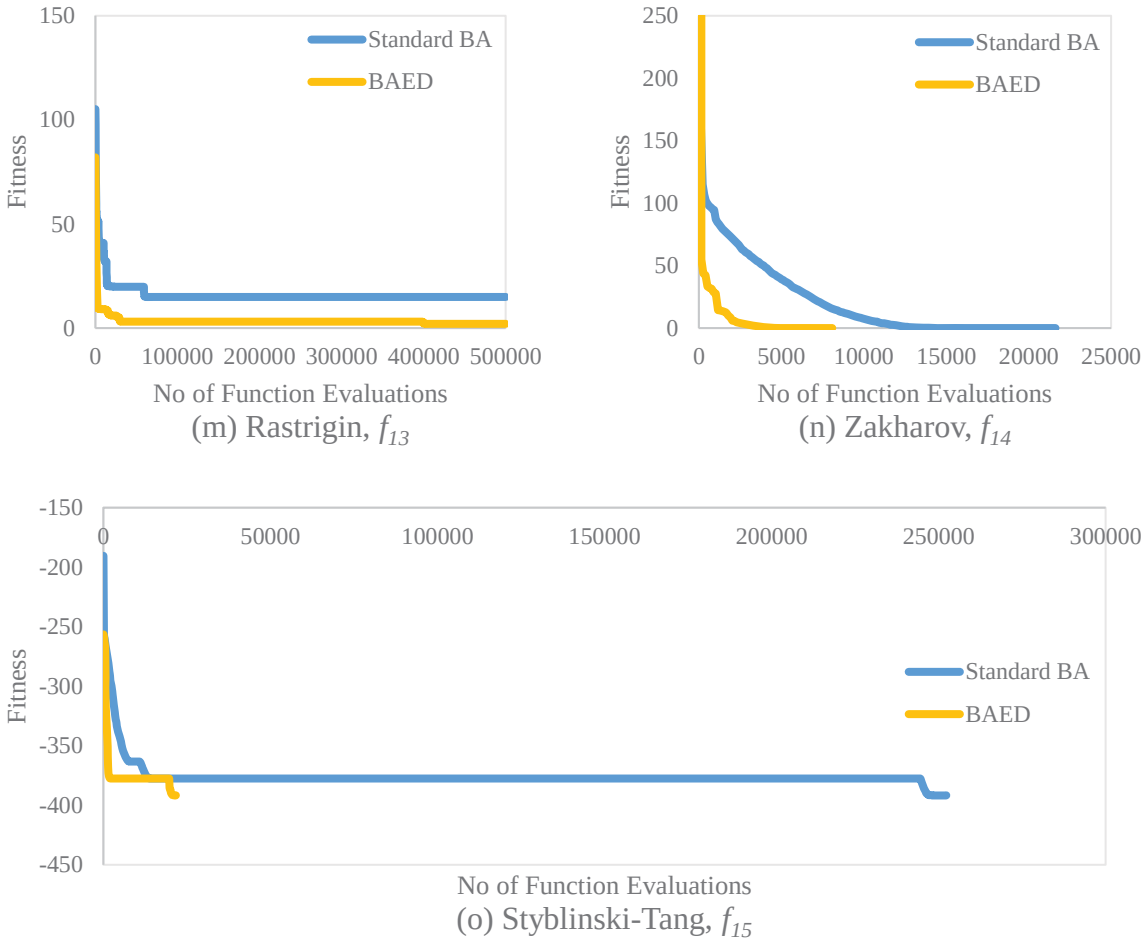


Figure 5.3: Sample graphs of convergence for standard Bees Algorithm and BAED (continued)

Furthermore, the performance of BAED was compared to SPSO2011 and qABC in similar benchmark functions. The results for 50 runs were tabulated in Table 5.3 and the *p*-values for accuracy and speed were recorded in Table 5.4. As seen in Table 5.3, the BAED outperforms the other algorithms in eleven out of fifteen benchmark functions. The BAED seems to work exceptionally well in low dimensional benchmark functions (f_1-f_6) compared to SPSO2011 and qABC especially in *Michalewicz*, *Schaffer*, and *Martin and Gaddy*. Tremendous improvement has been observed in the *Michalewicz* function where the median speed of BAED is 37 times and 9 times faster than SPSO2011 and qABC, respectively. Moreover, BAED maintains 100% success rate in all 50 runs in low dimensional functions (f_1-f_6).

Meanwhile, the performance of BAED in high dimensional benchmark functions (f_7-f_{15}) compared to SPSO2011 and qABC is excellent. In high dimensional unimodal functions (f_7-f_{10}), the BAED produces the best results in *Hypersphere* and *Axis*, qABC in *Rosenbrock* and BAED and qABC produce comparable performance but better than SPSO2011 in *Powell*. The results in high dimensional multimodal functions (f_{11}-f_{15}) show that BAED found that the optimum is faster in *Griewank, Zakharov* and *Styblinski-Tang*. However, in *Ackley* and *Rastrigin* the best performer is qABC with a higher number of success rate in the 50 runs conducted.

To further compare the performance of BAED, SPSO2011, and qABC, comparison graphs for 50 runs in each function have been constructed as in Figures 5.8 to 5.22. The blue box in some of the graphs represents a zoomed area for clarity purposes. For each function, the graph of speed is compared whether the algorithms have found the optimum value and inversely, the graph of accuracy is compared whether the algorithms could not find the optimum until the maximum number of iterations has elapsed. The definition of speed in this research is the number of function evaluation being called by the function and accuracy is defined as the difference between the best value found and the optimum value. The smaller the value of the speed of the algorithm, the faster the algorithm is in attaining the optimum value.

As can be inferred from the comparison graphs, in most functions the BAED shows good consistency in the 50 experimental runs. This characteristic indicates that the performance of the proposed algorithm is reliable and the improvement to the standard BA is significant and is not achieved by random coincidences.

Table 5.3: Performance comparison between BAED, SPSO2011 and qABC

Function	BAED			SPSO2011			qABC		
	Success	Accuracy	Speed	Success	Accuracy	Speed	Success	Accuracy	Speed
f_1	50	2.39E-04	**656**	50	5.42E-05	2800	38	8.82E-04	20223.5
f_2	50	2.58E-04	**530**	50	0.00E+00	3200	48	3.90E-04	2050
f_3	50	2.32E-04	**782**	50	5.00E-05	4000	50	2.16E-04	2450
f_4	50	1.10E-04	**782**	50	0.00E+00	3500	32	7.55E-04	102706
f_5	50	2.03E-04	**404**	50	3.60E-05	3100	50	4.31E-04	950
f_6	50	2.34E-04	**11748.5**	25	3.20E-03	433650	50	4.71E-04	111436.5
f_7	50	7.03E-04	**2231**	50	8.67E-05	10750	50	6.78E-04	92470
f_8	0	5.77E+00	500000	4	1.08E+01	500000	0	**9.27E-02**	500000
f_9	50	9.34E-04	98939	50	9.96E-05	88450	36	9.83E-04	269100.5
f_{10}	50	8.28E-04	**3176**	50	8.90E-05	13500	50	6.08E-04	117207.5
f_{11}	7	1.32E-01	500000	0	3.18E-01	500000	18	**1.63E-03**	500000
f_{12}	9	**1.51E-02**	500000	6	3.56E-02	500000	0	6.27E-02	500000
f_{13}	2	1.99E+00	500000	0	5.93E+00	500000	43	**7.80E-04**	**372419.5**
f_{14}	50	**8.75E-04**	**7964**	50	1.00E-04	44950	0	4.06E-02	500000
f_{15}	50	**5.76E-04**	**25104.5**	6	2.83E+01	500000	41	6.51E-04	144582

Table 5.4: Statistical comparison between BAED, SPSO2011 and qABC

Function	BAED			
	SPSO2011		qABC	
	Accuracy	Speed	Accuracy	Speed
f_1	1.0000	**0.0000**	**0.0000**	**0.0000**
f_2	1.0000	**0.0000**	0.0615	**0.0000**
f_3	1.0000	**0.0000**	1.0000	**0.0000**
f_4	1.0000	**0.0000**	**0.0000**	**0.0000**
f_5	1.0000	**0.0000**	1.0000	**0.0000**
f_6	0.0244	**0.0000**	1.0000	**0.0000**
f_7	1.0000	**0.0000**	1.0000	**0.0000**
f_8	**0.0035**	0.4902	**0.0000**	1.0000
f_9	1.0000	0.3421	**0.0004**	**0.0000**
f_{10}	1.0000	**0.0000**	1.0000	**0.0000**
f_{11}	**0.0000**	0.2301	**0.0000**	0.0854
f_{12}	**0.0001**	0.7114	**0.0000**	0.1211
f_{13}	**0.0000**	0.7279	**0.0000**	**0.0000**
f_{14}	1.0000	**0.0000**	**0.0000**	**0.0000**
f_{15}	**0.0000**	**0.0000**	0.5029	**0.0000**

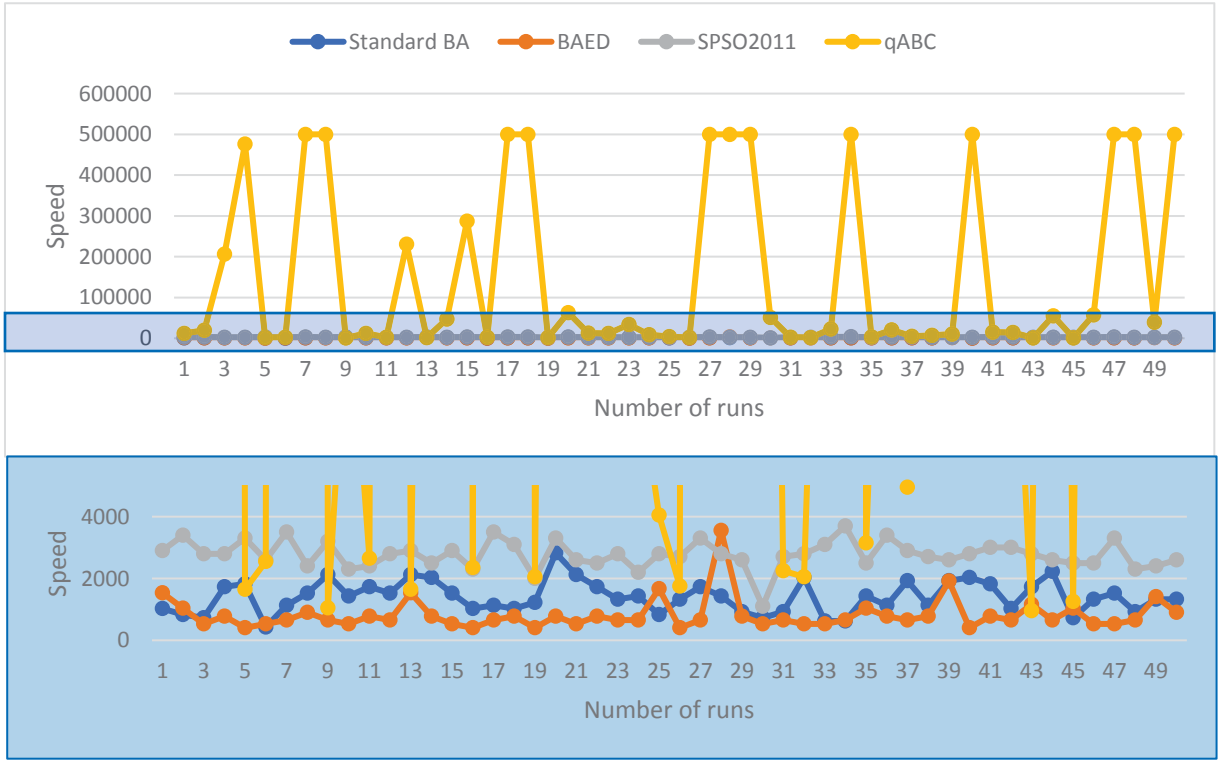


Figure 5.4: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_1

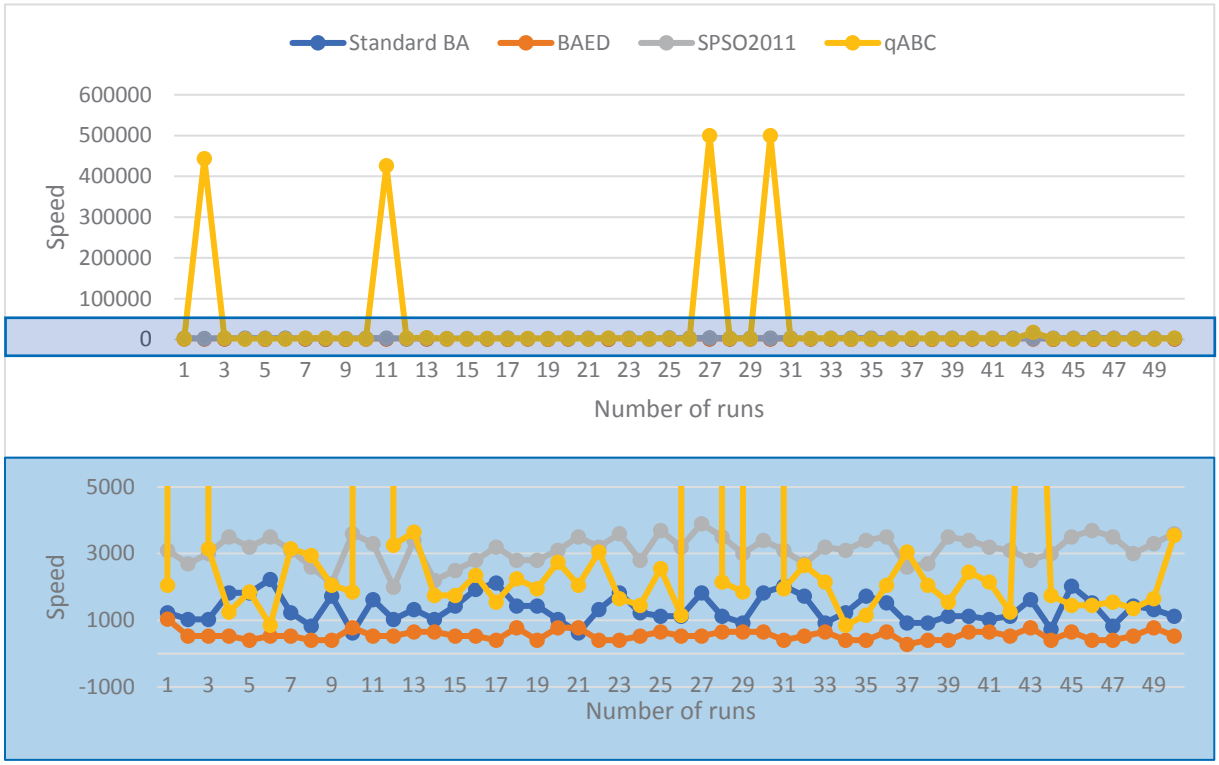


Figure 5.5: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_2

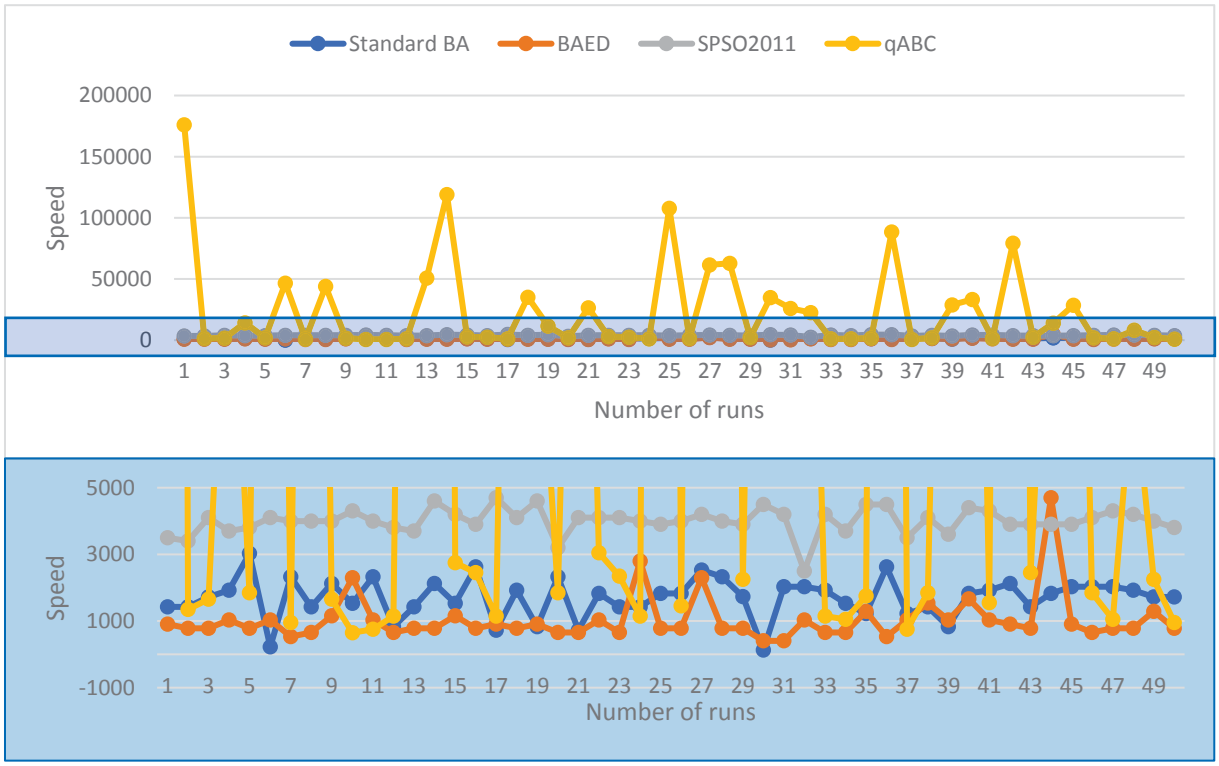


Figure 5.6: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_3

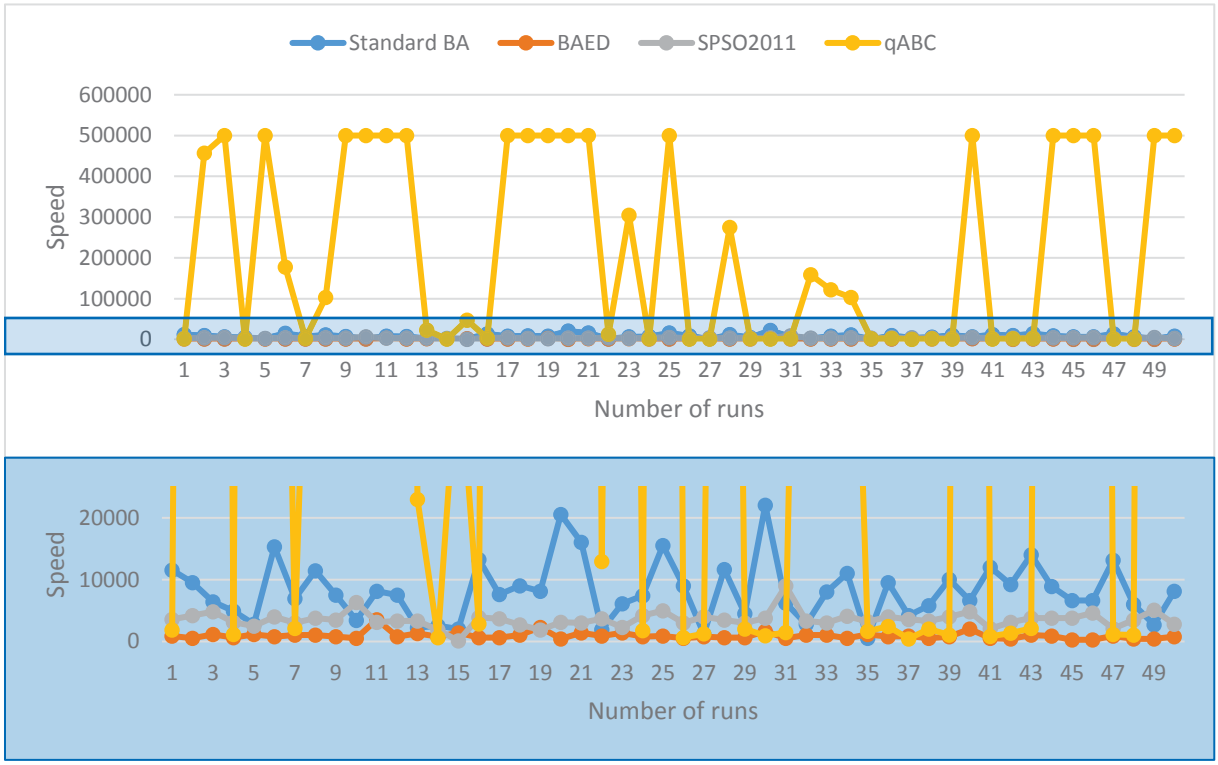


Figure 5.7: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_4

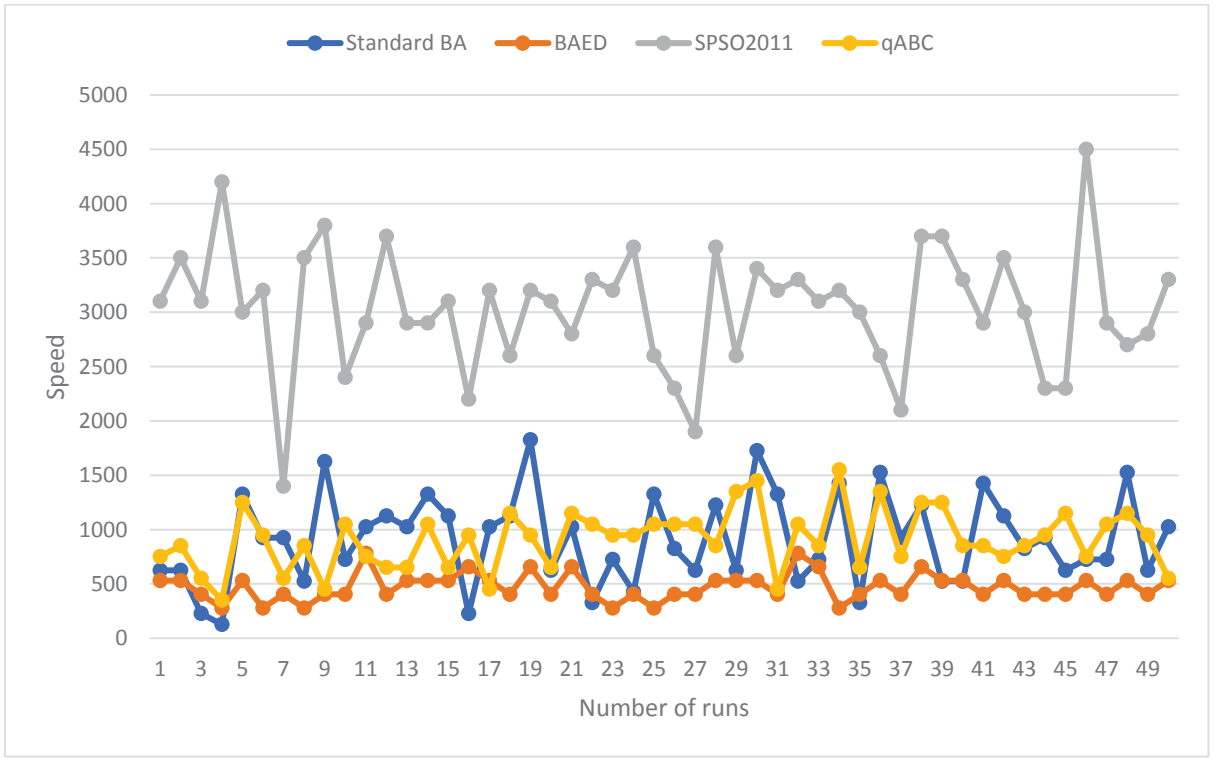


Figure 5.8: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_5

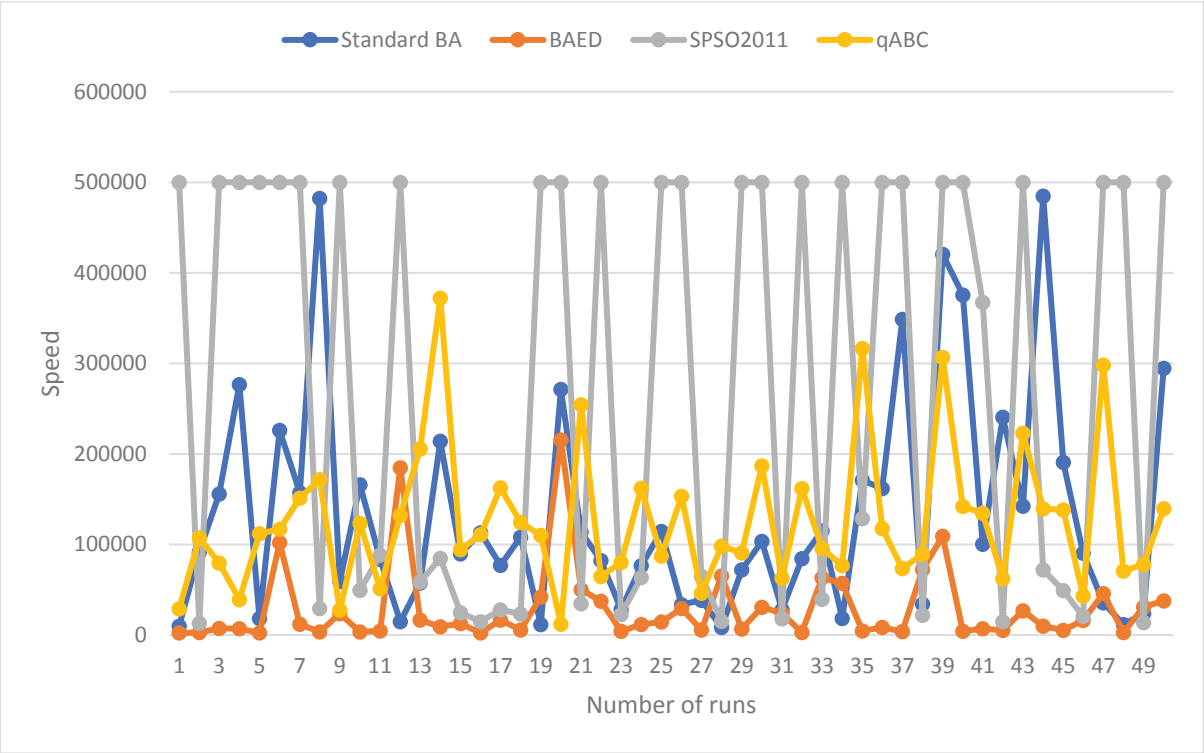


Figure 5.9: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_6

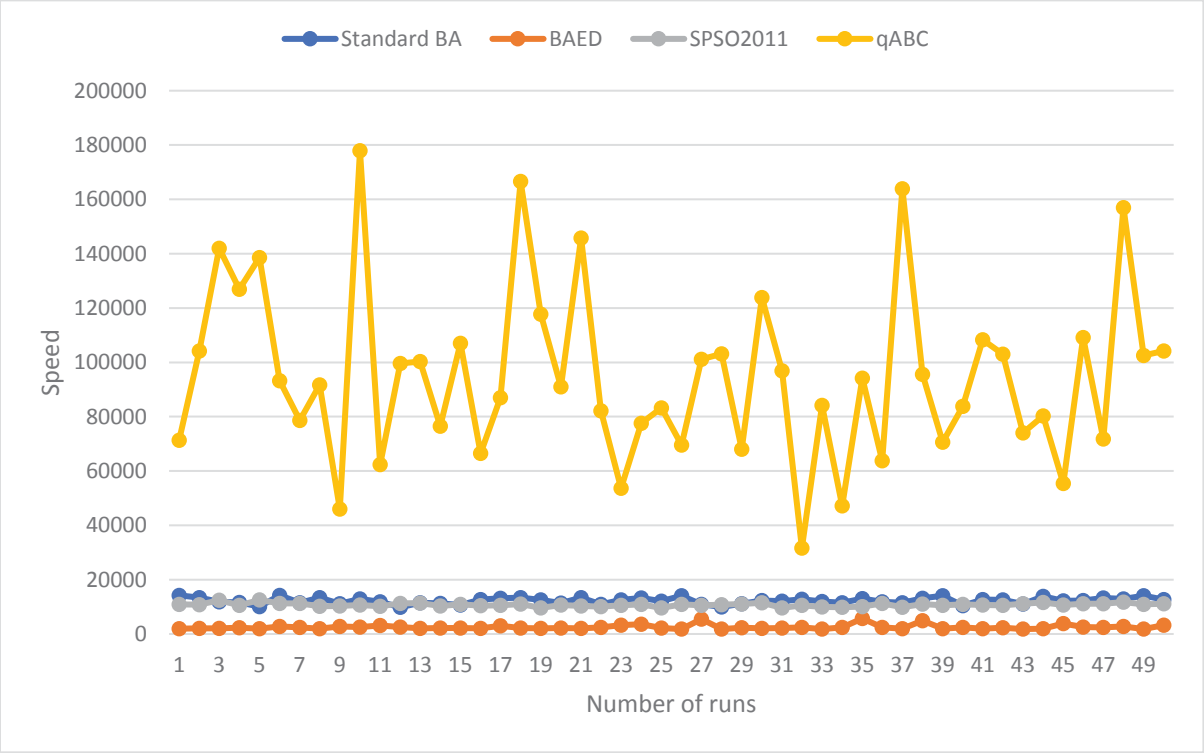


Figure 5.10: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_7

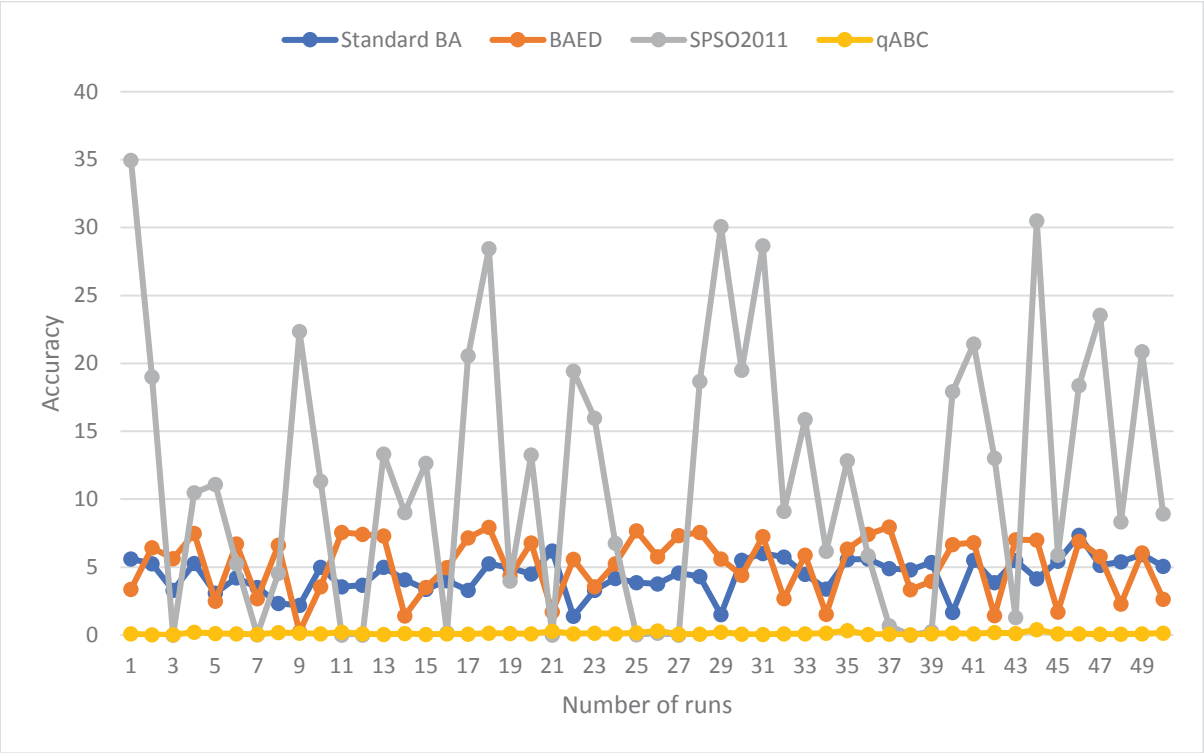


Figure 5.11: Comparison graph of accuracy performance between BAED, SPSO2011 and qABC for f_8

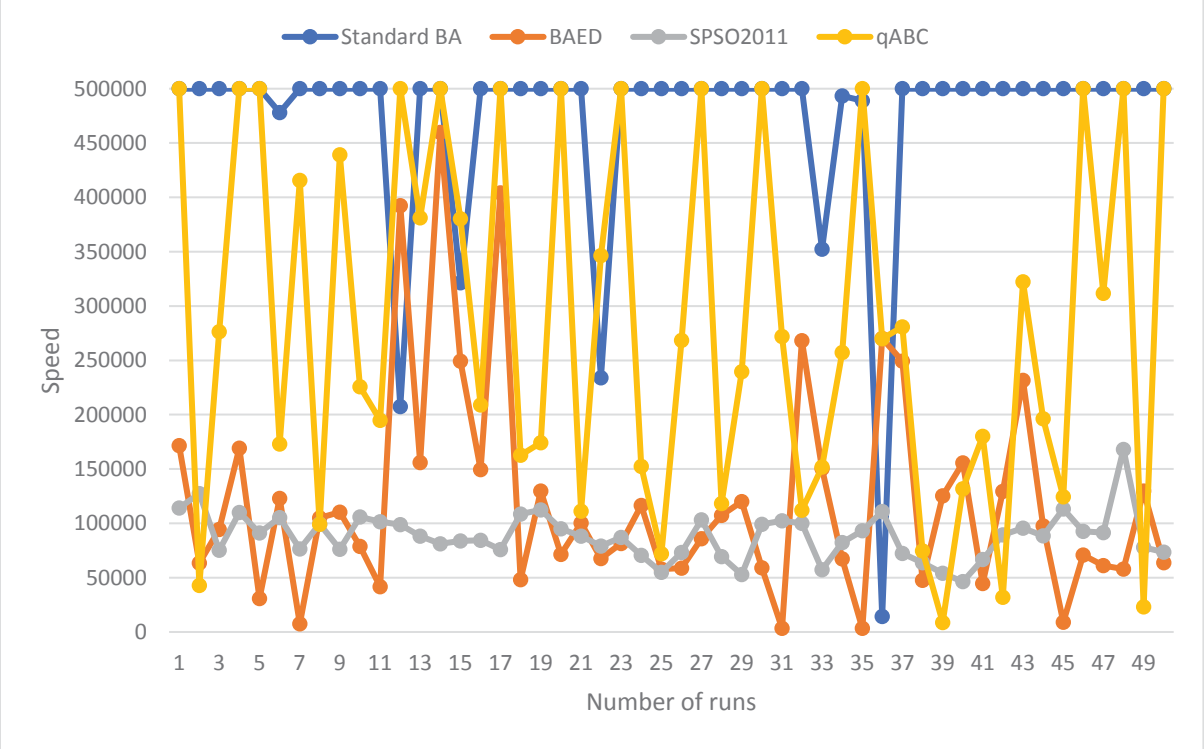


Figure 5.12: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_9

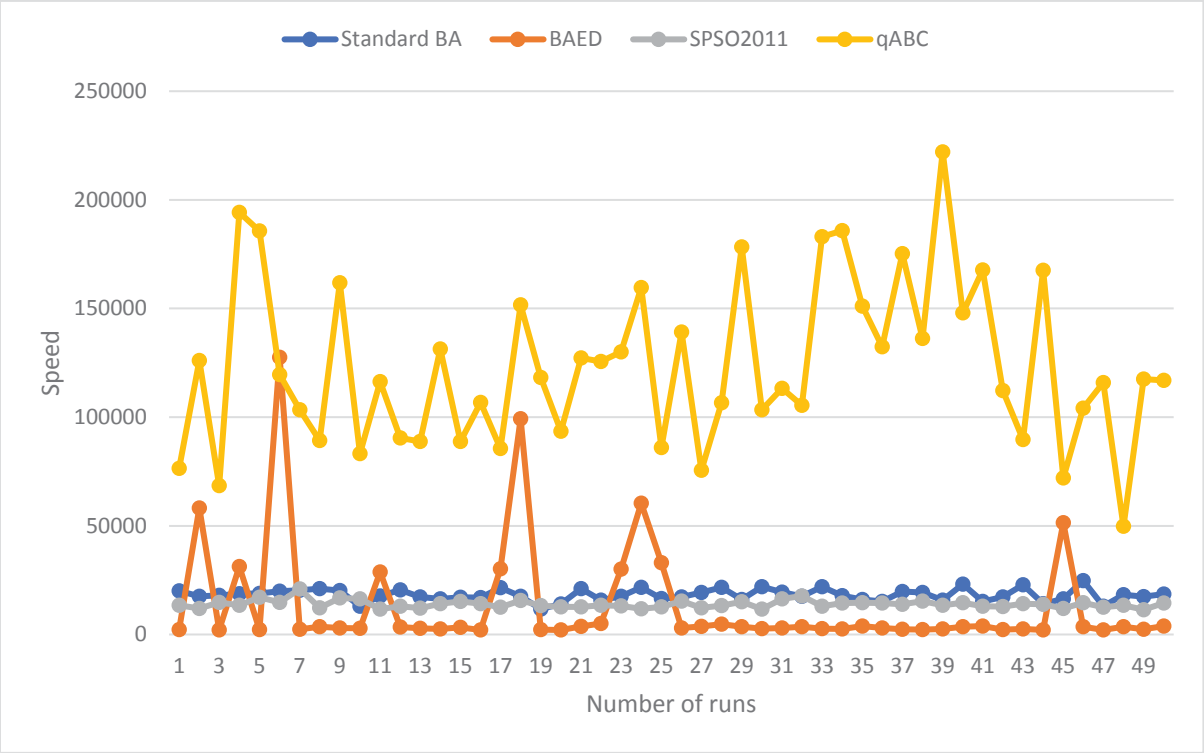


Figure 5.13: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_{10}

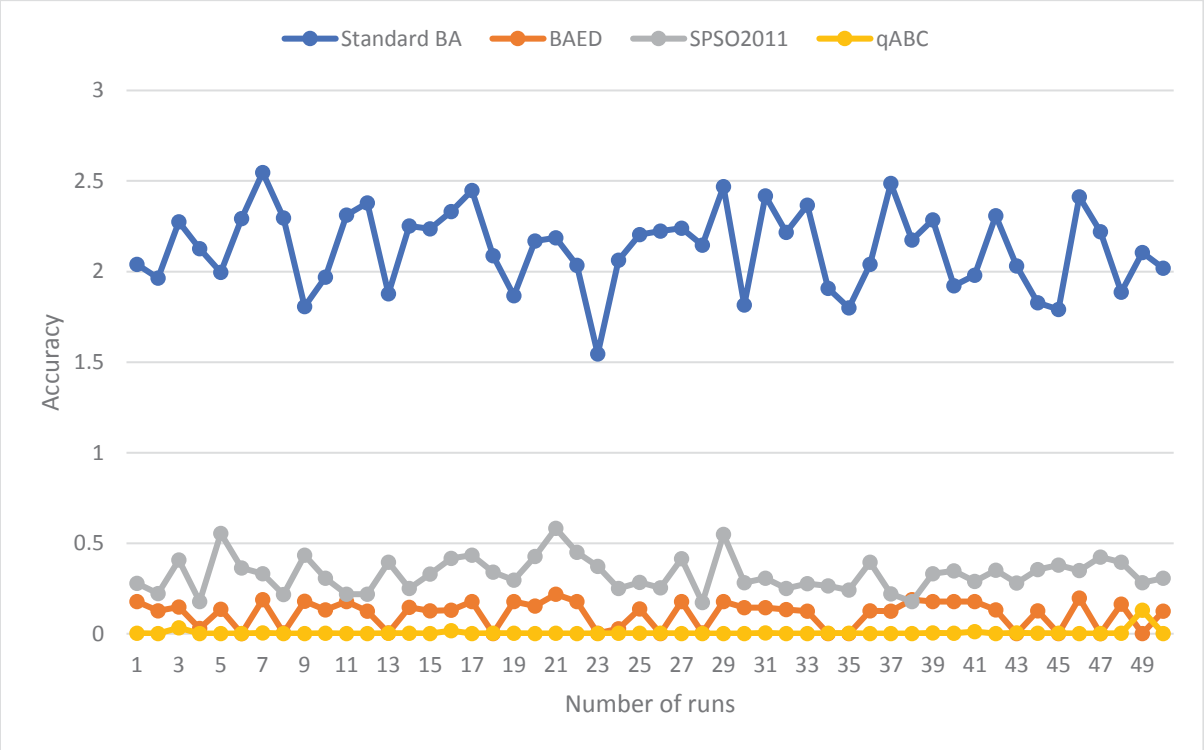


Figure 5.14: Comparison graph of accuracy performance between BAED, SPSO2011 and qABC for f_{11}

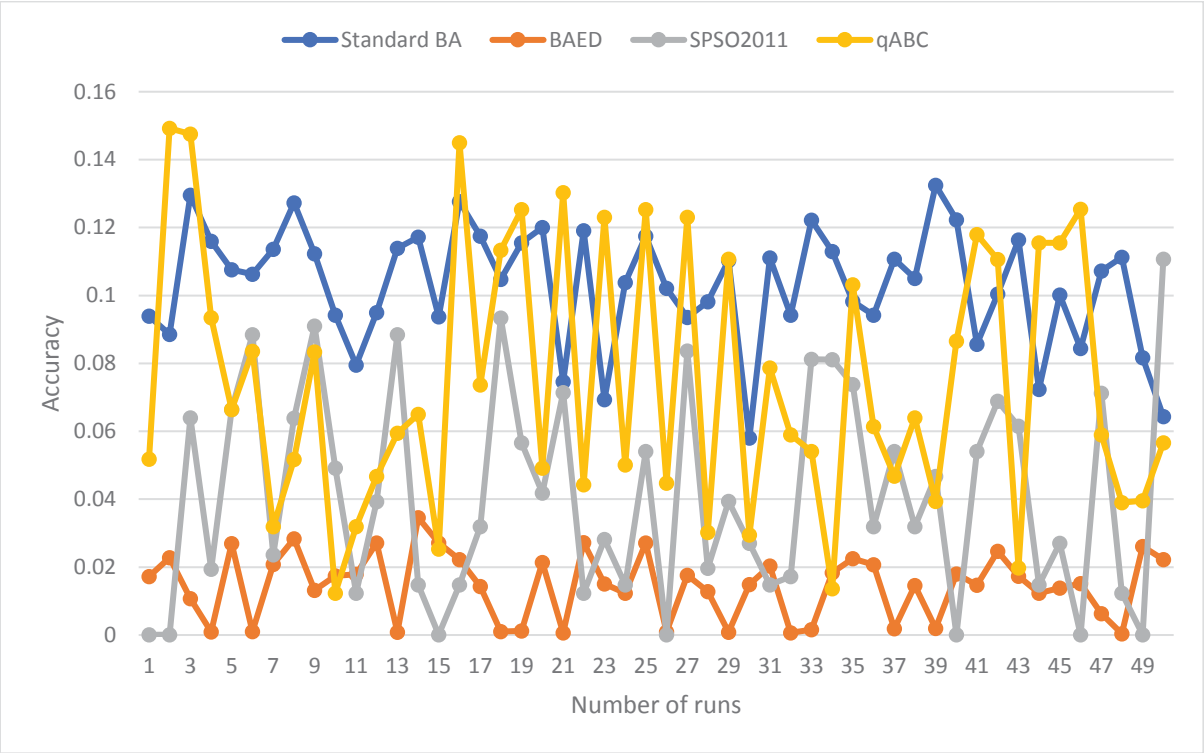


Figure 5.15: Comparison graph of accuracy performance between BAED, SPSO2011 and qABC for f_{12}

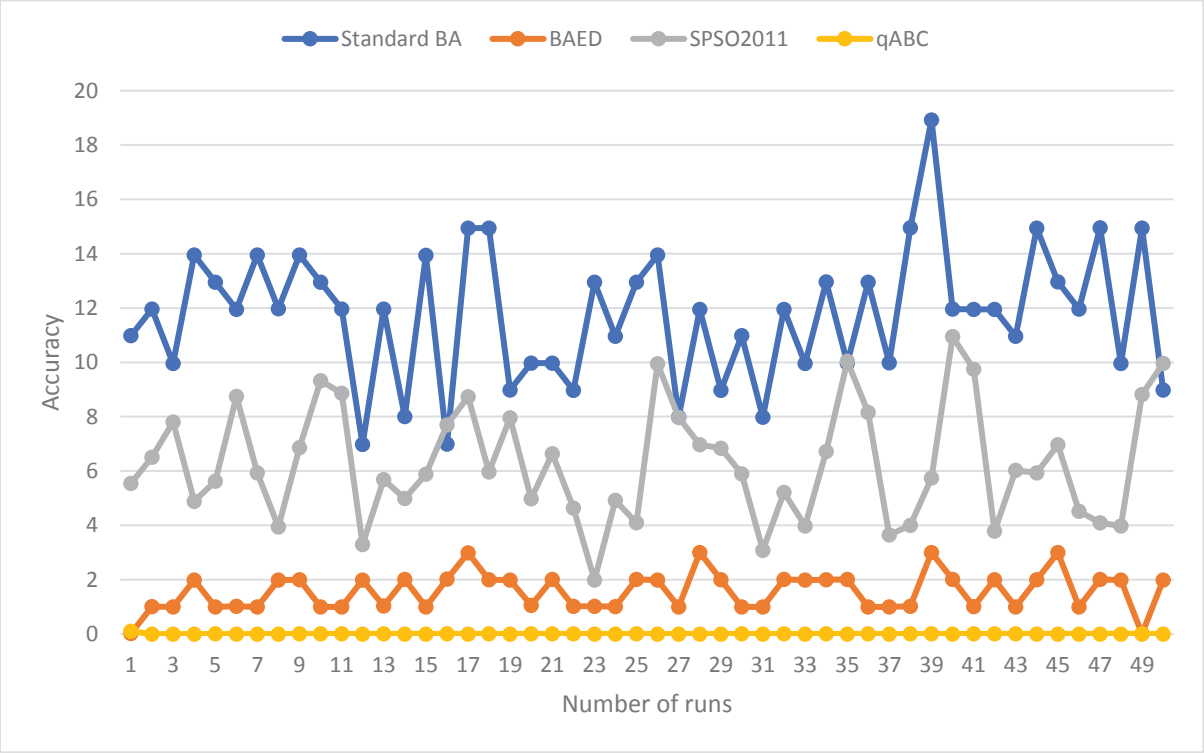


Figure 5.16: Comparison graph of accuracy performance between BAED, SPSO2011 and qABC for f_{13}

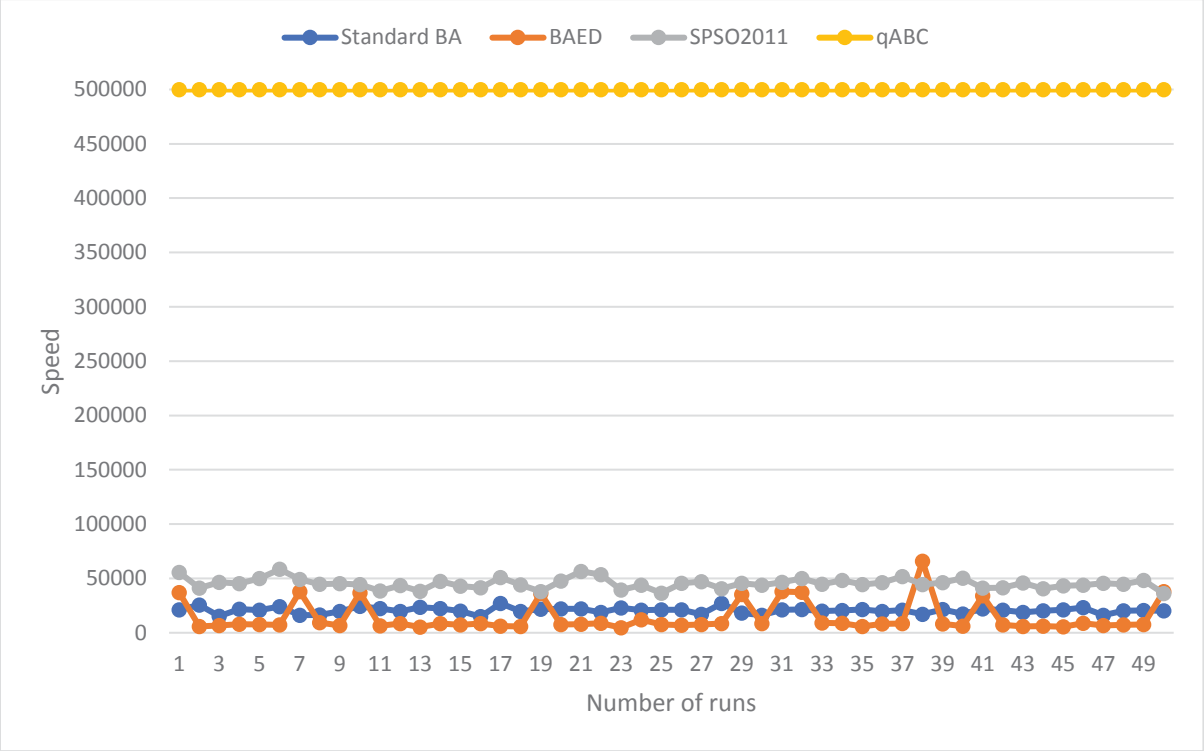


Figure 5.17: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_{14}

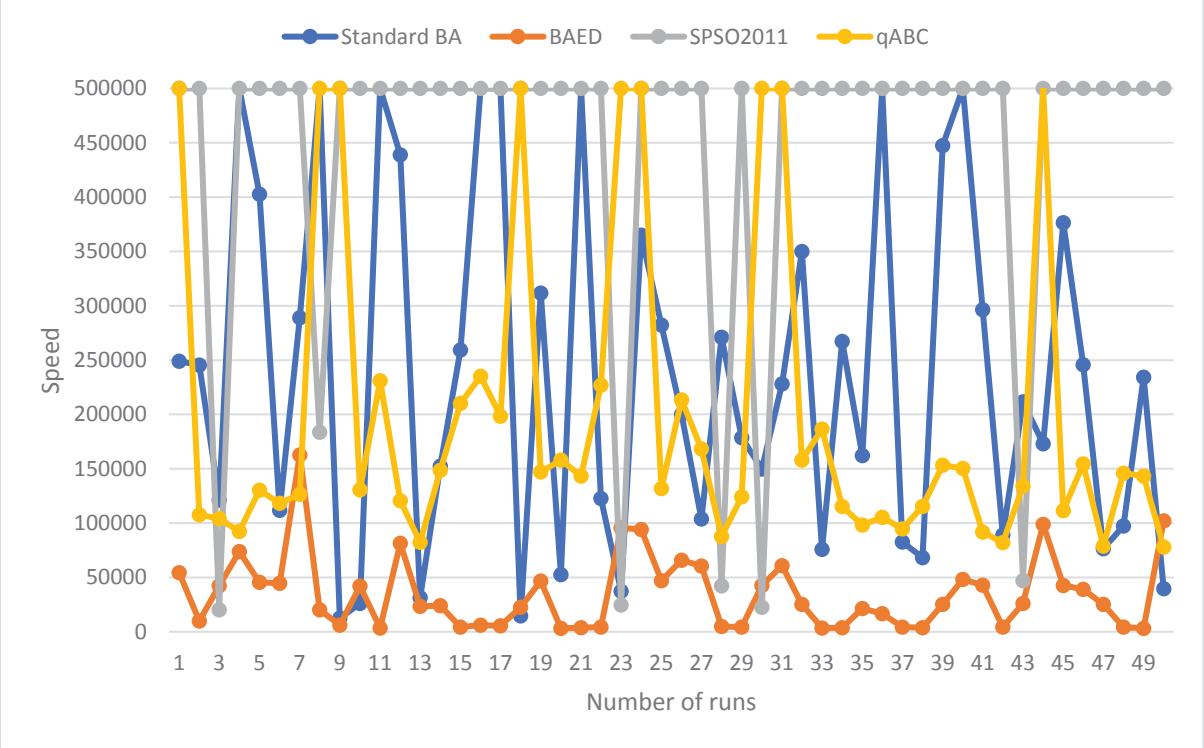


Figure 5.18: Comparison graph of speed performance between BAED, SPSO2011 and qABC for f_{15}

5.5 Engineering benchmark constrained and mechanical design problems

This section presents the performance of BAED on a set of constrained mechanical design problems as in Section 3.5 and Section 4.5. The parameter settings as previously adopted from Table 3.7 were used throughout this experiment and the stopping criteria and number of runs were set at 30,000 function evaluations and 30 times, respectively. Similarly, the Mann-Whitney significance test was used to determine whether the difference between the medians were statistically significant and the *p*-values were compared to significance level of 0.05. The results of BAED were compared to the standard BA and other selected algorithms reported in the literature. The best solutions and parameters obtained by BAED for the five constrained benchmark mechanical design problems are presented in Table 5.5. In addition, Table 5.6 shows the statistical results of BAED and the standard BA. The comparisons were made on the best solutions and medians obtained. The best results are written in boldface.

Table 5.5: Best results obtained by BAED for constrained benchmark mechanical design problems

	Three-truss bar	**Tension/compression spring**	**Pressure vessel**	**Welded beam**	**Speed reducer**
$f(x)$	263.8958485	0.01266783	6068.717201	1.725871682	2996.3563
x_1	0.788657254	0.051834644	0.8125	0.205939743	3.500002974
x_2	0.408298916	0.360192003	0.4375	3.469242949	0.700000011
x_3		11.089647	42.08819024	9.031924329	17
x_4			176.7638704	0.205949659	7.300228813
x_5					7.800000042
x_6					3.350216172
x_7					5.286690329

The performance of BAED is observed in Table 5.6. In all design problems tested, BAED produces better best solutions than standard BA. Nevertheless, BAED is significantly better in the three-truss bar, welded beam, and speed reducer problem in terms of the median data from 30 individual runs. According to Mann-Whitney significance test, there were no significant

differences between the solutions obtained from BAED and standard BA for tension/compression spring and pressure vessel.

Table 5.6: Comparison of the statistical results obtained from BAED and standard BA for constrained benchmark mechanical design problems

Problem		**Standard Bees Algorithm**	**BAED**
Three-bar truss	Best	263.8964263	**263.8958485**
	Median	263.9015189	**263.8979926**
	Mean	263.9032913	263.8985122
	SD	0.005504271	0.00204039
	Worst	263.919459	263.9057836
	Evaluations	30,000	30,000
Tension/compression spring	Best	0.012670733	**0.01266783**
	Median	0.012681849	0.012682025
	Mean	0.012760301	0.012683243
	SD	0.00017804	9.8593E-06
	Worst	0.013402018	0.012709633
	Evaluations	30,000	30,000
Pressure vessel	Best	6119.246703	**6068.717201**
	Median	6222.419644	6180.218102
	Mean	6259.109338	6280.003493
	SD	126.2859774	181.9933838
	Worst	6522.111587	6848.8576
	Evaluations	30,000	30,000
Welded beam	Best	1.73958454	**1.725871682**
	Median	1.767187634	**1.727991681**
	Mean	1.766361958	1.728883382
	SD	0.0150332	0.002299125
	Worst	1.810428812	1.735313946
	Evaluations	30,000	30,000
Speed reducer	Best	3006.240405	**2996.3563**
	Median	3038.251131	**2996.828155**
	Mean	3035.608189	2998.712697
	SD	10.27826125	6.212292616
	Worst	3050.226338	3030.379617
	Evaluations	30,000	30,000

Regarding the comparison with other algorithms from the literature, BAED was compared to previously solved problems as shown in Tables 5.7 to 5.11. As can be inferred from Table 5.7, the performance of BAED in terms of best solution for the three-truss bar problem is comparable to other algorithms. The difference in the best solutions found is relatively very small.

For tension/compression spring problem as shown in Table 5.8, the best solution is produced by ABC. The best solution of PSO-DE came second and BAED third. However, the best solution of these three top-ranked algorithms for this tension/compression spring problem can be said to be comparable to each other because differences were small. The performance of BAED in the remaining three mechanical design problems, pressure vessel, welded beam, and speed reducer were also found comparable to the other well-known algorithms.

Table 5.7: BAED comparison results for the three-truss bar optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
SC	263.895846	263.903356	263.969756	17,610
PSO-DE	263.89584338	263.89584338	263.89584338	17,600
Bees Algorithm	263.8964263	263.9032913	263.919459	30,000
BAED	263.8958485	263.8985122	263.9057836	30,000

Table 5.8: BAED comparison results for the tension/compression spring optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
GA3	0.0127048	0.0127690	0.0128220	900,000
GA4	0.0126810	0.0127420	0.0129730	80,000
PSO-DE	0.012665233	0.012665233	0.012665233	42,100
UPSOm	0.0131200	0.0229478	-	100,000
ABC	0.012665	0.012709	-	30,000
Bees Algorithm	0.012670733	0.012760301	0.013402018	30,000
BAED	0.01266783	0.012683243	0.012709633	30,000

Table 5.9: BAED comparison results for the pressure vessel optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
GA3	6288.7445	6293.8432	6308.4970	900,000
GA4	6059.9463	6177.2533	6469.3220	80,000
PSO-DE	6059.714335	6059.714335	6059.714335	42,100
UPSOm	6544.27	9032.55	-	100,000
ABC	6059.714736	6245.308144	-	30,000
Bees Algorithm	6119.246703	6259.109338	6522.111587	30,000
BAED	6068.717201	6280.003493	6848.8576	30,000

Table 5.10: BAED comparison results for the welded beam optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
GA3	1.748309	1.771973	1.785835	900,000
GA4	1.728226	1.792654	1.993408	80,000
PSO-DE	1.724852309	1.724852309	1.724852309	66,600
UPSOm	1.92199	2.83721		100,000
ABC	1.724852	1.741913	-	30,000
Bees Algorithm	1.73958454	1.766361958	1.810428812	30,000
BAED	1.725871682	1.728883382	1.735313946	30,000

Table 5.11: BAED comparison results for the speed reducer optimisation problem

Algorithm	Best	Mean	Worst	Evaluations
SC	2994.744241	3001.758264	3009.964736	54,456
PSO-DE	2996.348165	2996.348165	2996.348166	70,100
ABC	2997.058412	2997.058412	-	30,000
Bees Algorithm	3006.240405	3035.608189	3050.226338	30,000
BAED	2996.3563	2998.712697	3030.379617	30,000

5.6 Summary

This chapter demonstrated the validity of a new variant of BA called BAED that utilises the EDA method and analyses its performance on continuous global optimisation problems. Experimental results to compare the performance of this new variant on a set of optimisation problems in continuous domains were compared with those for the standard BA and other well-known algorithms as well. In light of the results, it can be concluded that BAED had proved to be capable of improving on the standard BA and achieving significantly better performance than those obtained by SPSO2011 and qABC for most of the benchmark functions. Specifically, BAED discovered the optimal solution with the least number of evaluations in fourteen out of fifteen cases compared to the standard BA, and eleven out of fifteen functions compared to SPSO2011 and qABC. In addition, BAED produced the best solution in all five constrained mechanical design problems compared to the standard BA. Evaluation against

other algorithms from the literature showed that BAED was comparable in terms of performance in the constrained design problems.

CHAPTER 6

CONCLUSION

This chapter summarises the main contributions of this research and the conclusions reached and makes suggestions for future work.

6.1 Summary

The objectives specified in Chapter 1 have all been accomplished.

This thesis proposed three different enhanced versions of the Bees Algorithm to improve its performance in terms of accuracy and convergence speed in handling global continuous optimisation problems. The proposed algorithms were tested on unconstrained numerical benchmark functions with known solutions and constrained mechanical design benchmark problems. All simulation results were provided in related chapters. The summary and conclusions reached were as follows:

i. The intensification of the local search procedure by applying a direct search method called the Hooke and Jeeves' method to yield the Bees Algorithm with Hooke and Jeeves (BA-HJ) were established. In this approach, the HJ used the location of the best-so-far solution as the starting point and the new location obtained by this method was recorded as the new elite sites for the current population. Then, the new elite sites will undergo the same procedure as in the standard BA. The BA-HJ performed better than the standard BA in most of the benchmark functions and mechanical design problems. Comparable performance was also achieved between the proposed algorithm and other

algorithms found in the literature. The addition of a pattern move element in the BA-HJ helped to increase the directional search in the algorithm thus improving its success rate, convergence speed and accuracy. This conclusion addresses objective (i).

ii. The second improvement was a method to replace the random generation of the recruited bee's deployment with chaotic sequences. A well-known logistic map was used in this approach to produce chaotic sequences in the proposed algorithm called Bees Algorithm with Chaos (ChaosBA). The ChaosBA began with the whole population used as the scout bees for initialisation to get as many points as possible to cover the search space. The ChaosBA continued with the similar procedures of standard BA until the elite and best sites had been selected. At this stage, the points of the current elite and best sites were utilised as the initial points for the chaotic sequences to generate new points for the local and global search procedures to complete the cycle of the algorithm. The ChaosBA was tested on a set of benchmark functions and compared with the standard BA, SPSO2011, and qABC. The ChaosBA performed better in most of the functions and produced a performance in the mechanical design problems comparable with that by other algorithms in the literature. The chaotic process of generating new candidate solutions showed some improvements to the performance of the Bees Algorithm. This conclusion addresses objective (ii).

iii. Finally, the Bees Algorithm with Estimation Distribution (BAED) was introduced by applying the probabilistic method in EDA using the information of the current best solutions to guide the next search. In BAED, after the population is evaluated and ranked, the best sites are used to generate new candidate solutions by sampling the distribution induced by the Gaussian probabilistic model. The candidate solutions are then evaluated and merged back into the old population to be ranked once again. Finally, the ranked population is truncated according to the parameter of BAED. The

newly truncated population now has new best sites to be used in the remaining search procedures of the algorithm. This approach showed tremendous improvement in the simulation test using the same benchmark functions and mechanical design problems as in Chapter 3 when compared to the standard BA. Furthermore, the performance of BAED was also found to be comparable to other well-known algorithms found in the literature in the engineering mechanical design problems. Therefore, it was evident that utilising the information of the best solutions to guide the search had a good impact on the performance of the Bees Algorithm. This conclusion addresses objective (iii).

6.2 Contributions

This research introduced new enhancements to the Bees Algorithm to advance the ability of the algorithm in solving global continuous optimisation problems. The main contributions are as follows:

i. The development of BA-HJ enables the intensification of the local search procedure by incorporating a direct search method to strengthen the exploitation policy of the algorithm. The proposed BA-HJ demonstrated strong competitive results in terms of its success at locating the optimum solution, convergence speed, and accuracy when compared with the standard BA, SPSO2011 and qABC. The BA-HJ outperformed the standard BA in thirteen out of fifteen benchmark functions and is more effective in eleven out of fifteen benchmark functions when compared to SPSO2011 and qABC. Furthermore, BA-HJ have shown the best solution in all five constrained mechanical design problems compared to that of the standard BA in terms of the performance against other algorithms from the literature.

ii. The development of ChaosBA provides the recruited bee's deployment to follow the unique traits of chaotic sequences in order to enhance the exploitation and exploration capabilities of the algorithm to reach the global optimum. The ChaosBA performed effectively in the unconstrained benchmark functions compared to the standard BA. The experimental results revealed that the proposed algorithm is an efficient and effective algorithm than that of the standard BA including the difficult multimodal functions, such as *Rosenbrock*, *Powell*, *Ackley*, and *Rastrigin*. Altogether, ChaosBA outperformed the standard BA in twelve out of fifteen benchmark functions. The results obtained with another two state-of-the-art algorithms have also shown evidence that the ChaosBA performance is significantly better in eleven out of fifteen test functions. Also, ChaosBA is able to determine the best solution in all the constrained design problems and performs effectively for any engineering applications, for instance the speed reducer problem, when compared to the standard BA.

iii. The development of BAED enables the algorithm to take advantage of available information from the current population to produce new candidate solutions using a probabilistic method combined with existing strategies of the algorithm to improve its convergence speed and accuracy. BAED discovered the optimal solution with the least number of evaluations in fourteen out of fifteen cases compared to the standard BA, and eleven out of fifteen functions compared to SPSO2011 and qABC. In addition, BAED produced the best solution in all the five constrained mechanical design problems compared to the standard BA, which is comparable in terms of performance against other algorithms from the literature.

6.3 Future work

This section suggests promising new directions for further research with the aim in enhancing the algorithms. The future works are given as follow:

i. The proposed BA-HJ which has focussed on intensifying the local search via the direct search method has shown tremendous improvement. However, the incorporation of the direct search was carried out only in the local search phase of the algorithm. In the future, different phase of the algorithm can be targeted to be incorporated with the direct search. Combination with other types of direct search could also be an area to explore.

ii. Changing the way, the Bees algorithm works in terms of the deployment method from random to chaotic sequences enables the proposed algorithm to show some excellent results in this study. However, the proposed ChaosBA only utilises one type of chaotic map in the process. Hence, it is worth investigating different types of chaotic map on the performance of the algorithm. Amalgamation of various chaotic map in different phases of the algorithm would be a good concept to delve.

iii. This study found that capitalising the current information of the population enables the algorithm to produce better candidate solutions during the search. The proposed BAED has shown outstanding performance compared to the standard BA. Since only one type of distribution is used in the probabilistic model of this algorithm, future work on other distribution such as the mixed Gaussian would be an interesting idea to be explored.

iv. Based on the results from this study, the proposed algorithms through BA-HJ and BAED exhibit excellent convergence speed and accuracy. Thus, combining these two concepts in a new approach would be an interesting attempt to further improve the capability of the algorithm in handling complex optimisation problems in the future.

REFERENCES

Abbas, Abbas Khudhair, Ahmed Tariq Sadeq, Educational Continues, and Computer Science. 2014. "Database Clustering Using Intelligent Techniques." *Journal of Al-Nahrain University* 17(3): 195–203.

Abd Elazim, S. M., and E. S. Ali. 2016. "Optimal Power System Stabilizers Design via Cuckoo Search Algorithm." *International Journal of Electrical Power and Energy Systems* 75: 99–107.

Abdullah, Salwani, and Malek Alzaqebah. 2013. "A Hybrid Self-Adaptive Bees Algorithm for Examination Timetabling Problems." *Applied Soft Computing Journal*.

Ahmed, Jubaer, and Zainal Salam. 2014. "A Maximum Power Point Tracking (MPPT) for PV System Using Cuckoo Search with Partial Shading Capability." *Applied Energy* 119: 118–30.

Ahn, Chang Wook, Jinung An, and Jae Chern Yoo. 2012. "Estimation of Particle Swarm Distribution Algorithms: Combining the Benefits of PSO and EDAs." *Information Sciences* 192: 109–19.

Akay, Bahriye, and Dervis Karaboga. 2012. "Artificial Bee Colony Algorithm for Large-Scale Problems and Engineering Design Optimization." *Journal of Intelligent Manufacturing* 23(4): 1001–14.

Al-araji, Ahmed S, and Noor Q. Yousif. 2017. "Design of a Nonlinear Controller for Wheeled Mobile Robot Based on Cognitive On-Line Hybrid Bees-PSO Optimization Algorithm." In *1st International Conference on Recent Trends of Engineering Sciences and Sustainability*,.

Al-Araji, Ahmed Sabah. 2016. "Development of a Swing-Tracking Sliding Mode Controller Design for Nonlinear Inverted Pendulum System via Bees-Slice Genetic Algorithm." *Eng. & Tech. Journal* 34(15): 2897–2910.

Al-obaidi, Ahmed T Sadiq, Hasanen S Abdullah, and Zied O Ahmed. 2018. "Meerkat Clan Algorithm : A New Swarm Intelligence Algorithm." *Indonesian Journal of Electrical Engineering and Computer Science* 10(1): 354–60.

Alfi, Alireza, and A. Khosravi. 2012. "Constrained Nonlinear Optimal Control via a Hybrid BA-SD." *International Journal of Engineering-Transactions C: Aspects* 25(3): 197–204.

Amsaleka, R, and M Latha. 2014. "A Optimally Enhanced Fuzzy K-C Means (Oefkcm) For Clustering Algorithm Medical Image Segmentation." 3(3): 5678–82.

Ang, M. C., D. T. Pham, Anthony J. Soroka, and Kok W. Ng. 2010. "PCB Assembly Optimisation Using the Bees Algorithm Enhanced with TRIZ Operators." In *IECON Proceedings (Industrial Electronics Conference)*, 2708–13.

Ang, M C, D T Pham, and K W Ng. 2009. "Minimum-Time Motion Planning for a Robot Arm Using the Bees Algorithm." In *7th IEEE International Conference on Industrial Informatics (INDIN 2009)*, 487–92.

Arora, Jasbir Singh. 1989. *Introduction to Optimun Design*. New York: McGraw-Hill.

Askarzadeh, Alireza, and Alireza Rezazadeh. 2013. “A New Heuristic Optimization Algorithm for Modeling of Proton Exchange Membrane Fuel Cell: Bird Mating Optimizer.” *International Journal of Energy Research* 37(10): 1196–1204.

Assareh, Ehsanolah, and Mojtaba Biglari. 2016. “A Novel Approach to Capture the Maximum Power Generation from Wind Turbines Using Hybrid MLP Neural Network and Bees Algorithm (HNNBA).” *IETE Journal of Research* 62(3): 368–78.

Aziz, Mohamed Abd El, and Aboul Ella Hassanien. 2016. “Modified Cuckoo Search Algorithm with Rough Sets for Feature Selection.” *Neural Computing and Applications*: 1–10.

Bahamish, Hesham Awadh A., Rosni Abdullah, and Rosalina Abdul Salam. 2008. “Protein Conformational Search Using Bees Algorithm.” In *2nd Asia International Conference on Modelling and Simulation, AICMS 2008*, Kuala Lumpur, Malaysia, 911–16.

Beni, Gerardo, and Jing Wang. 1993. “Swarm Intelligence in Cellular Robotic Systems.” *Robots and Biological Systems: Towards a New Bionics?* (2): 703–12.

Bhandari, A. K., V. Soni, A. Kumar, and G. K. Singh. 2014. “Cuckoo Search Algorithm Based Satellite Image Contrast and Brightness Enhancement Using DWT-SVD.” *ISA Transactions* 53(4): 1286–96.

Bilchev, G., and I. C. Parmee. 1995. “The Ant Colony Metaphor for Searching Continuous Design Spaces.” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 993: 25–39.

Bosman, Peter A N, and Dirk Thierens. 2000. “Expanding from Discrete to Continuous Estimation of Distribution Algorithms : The IDEA.” In *Parallel Problem Solving from Nature PPSN VI: 6th International Conference Paris, France, September 18--20, 2000 Proceedings*, eds. Marc Schoenauer et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 767-776.

Caponetto, Riccardo, Luigi Fortuna, Stefano Fazzino, and Maria Gabriella Xibilia. 2003. “Chaotic Sequences to Improve the Performance of Evolutionary Algorithms.” *IEEE Transactions on Evolutionary Computation* 7(3): 289–304.

Chatterjee, Sankhadeep et al. 2017. “Cuckoo Search Coupled Artificial Neural Network in Detection of Chronic Kidney Disease.” *2017 1st International Conference on Electronics, Materials Engineering and Nano-Technology (IEMENTech)*: 1–4.

Chu, Shu-Chuan, Pei-wei Tsai, and Jeng-Shyang Pan. 2006. “Cat Swarm Optimization.” In *PRICAI 2006: Trends in Artificial Intelligence. PRICAI 2006. Lecture Notes in Computer Science, Vol 4099*, eds. Q. Yang and G. Webb. Springer, Berlin, Heidelberg, 854–58.

Cobos, Carlos et al. 2014. “Clustering of Web Search Results Based on the Cuckoo Search Algorithm and Balanced Bayesian Information Criterion.” *Information Sciences* 281: 248–64.

Coello Coello, Carlos A. 2000. “Use of a Self-Adaptive Penalty Approach for Engineering Optimization Problems.” *Computers in Industry* 41(2): 113–27.

Coello Coello, Carlos A., and Efrén Mezura Montes. 2002. “Constraint-Handling in Genetic Algorithms through the Use of Dominance-Based Tournament Selection.” *Advanced Engineering Informatics* 16(3): 193–203.

Das, Sanjoy, Bijaya K. Panigrahi, and Shyam S. Pattnaik. 2010. “Nature Inspired Methods for Multi-Objective Optimization.” In *Handbook of Research on Machine Learning Applications and Trends*, eds. E. Olivas et al. Hershey, PA: IGI Global, 95–108.

Dereli, Türkay, and Gülesin Sena Das. 2011. “A Hybrid ‘bee(s) Algorithm’ for Solving Container Loading Problems.” *Applied Soft Computing Journal* 11(2): 2854–62.

Devabalaji, K. R., T. Yuvaraj, and K. Ravi. 2015. “An Efficient Method for Solving the Optimal Sitting and Sizing Problem of Capacitor Banks Based on Cuckoo Search Algorithm.” *Ain Shams Engineering Journal*.

Dinh, Bach Hoang, Thang Trung Nguyen, and Dieu Ngoc Vo. 2016. “Adaptive Cuckoo Search Algorithm for Short-Term Fixed-Head Hydrothermal Scheduling Problem with Reservoir Volume Constraints.” *International Journal of Grid and Distributed Computing* 9(5): 191–204.

Dorigo, M., and G. Di Caro. 1999. “Ant Colony Optimization: A New Meta-Heuristic.” *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)* 2: 1470–77.

Dorigo, M, V Maniezzo, and a Colorni. 1991. “The Ant System: An Autocatalytic Optimizing Process.” *TR91-016, Politecnico di Milano*: 1–21.

Esfandiari, Ahmad. 2014. “Cuckoo Optimization Algorithm in Cutting Conditions During Machining.” *Journal of Advances in Computer Research* 5(2): 45–57.

Farajvand, Mohammad, Vahid Kiarostami, Mehran Davallo, and Abdolmohammad Ghaedi. 2018. “Optimization of Solvent Terminated Dispersive Liquid – Liquid Microextraction of Copper Ions in Water and Food Samples Using Artificial Neural Networks Coupled Bees Algorithm.” *Bulletin of Environmental Contamination and Toxicology* 100(3): 402–8.

Gandomi, A H, X Yang, S Talatahari, and A H Alavi. 2013. “Firefly Algorithm with Chaos.” *Communications in Nonlinear Science and Numerical Simulation* 18(1): 89–98.

Gandomi, Amir H., and Xin She Yang. 2014. “Chaotic Bat Algorithm.” *Journal of Computational Science* 5(2): 224–32.

Gandomi, Amir Hossein, and Amir Hossein Alavi. 2012. “Krill Herd: A New Bio-Inspired Optimization Algorithm.” *Communications in Nonlinear Science and Numerical Simulation* 17(12): 4831–45.

Garcia, F.J.M., and J.A.M. Perez. 2008. Technical Report 3, Documentos de Trabajo del DEIOC, Department of Statistics, O.R. and Computing *Jumping Frogs Optimization: A New Swarm Method for Discrete Optimization*. Tenerife, Spain.

Ghiasi, Mohammad et al. 2017. “A New Prediction Model of Electricity Load Based on Hybrid Forecast Engine.” *International Journal of Ambient Energy* 0750: 1–8.

Gholipour, Reza, Alireza Khosravi, and Hamed Mojallali. 2015. “Multi-Objective Optimal Backstepping Controller Design for Chaos Control in a Rod-Type Plasma Torch System Using Bees Algorithm.” *Applied Mathematical Modelling* 39: 4432–44.

Giridhar, Munigoti S., S. Sivanagaraju, Chintalapudi V. Suresh, and P. Umapathi Reddy. 2017. “Analyzing the Multi Objective Analytical Aspects of Distribution Systems with Multiple Multi-Type Compensators Using Modified Cuckoo Search Algorithm.”

International Journal of Parallel, Emergent and Distributed Systems 32(6): 549–71.

Glover, Fred. 1986. "Future Paths for Integer Programming and Links to Artificial Intelligence." *Computers and Operations Research* 13(5): 533–49.

Glover, Fred, and G.A. Kochenberger. 2003. *Handbook of Metaheuristics*. Springer New York.

Gogna, Anupriya, and Akash Tayal. 2013. "Metaheuristics: Review and Application." *Journal of Experimental and Theoretical Artificial Intelligence* 25(4): 503–26.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Massachusetts, USA: Addison-Wesley Publishing Company, Inc.

Guney, Kerim, and Murat Onay. 2007. "Amplitude-Only Pattern Nulling of Linear Antenna Arrays With the Use of Bees Algorithm." *Progress In Electromagnetics Research, PIER 70*: 21–36.

Guney, Kerim, and Murat Onay. 2008. "Bees Algorithm for Design of Dual-beam Linear Antenna Arrays with Digital Attenuators and Digital Phase Shifters." *International Journal of RF and Microwave Computer-Aided Engineering* 18(4): 337–47.

Haj Darwish, Ahmed, Abdulkader Joukhadar, and Mariam Kashkash. 2018. "Using the Bees Algorithm for Wheeled Mobile Robot Path Planning in an Indoor Dynamic Environment." *Cogent Engineering* 5(1).

Hauschild, Mark, and Martin Pelikan. 2011. "An Introduction and Survey of Estimation of Distribution Algorithms." *Swarm and Evolutionary Computation* 1(3): 111–28.

Havens, Timothy C., Christopher J. Spain, Nathan G. Salmon, and James M. Keller. 2008. "Roach Infestation Optimization." *2008 IEEE Swarm Intelligence Symposium, SIS 2008*.

Hooke, Robert, and T. A. Jeeves. 1961. "'Direct Search''' Solution of Numerical and Statistical Problems.'" *Journal of the ACM* 8(2): 212–29.

Hussein, Wasim A., Shahnorbanun Sahran, and Siti Norul Huda Sheikh Abdullah. 2014. "Patch-Levy-Based Initialization Algorithm for Bees Algorithm." *Applied Soft Computing* 23: 104–21.

Hussein, Wasim A, Shahnorbanun Sahran, Siti Norul, and Huda Sheikh. 2015. "An Improved Bees Algorithm for Real Parameter Optimization." *International Journal of Advanced Computer Science and Applications* 6(10): 23–39.

Ismkhan, Hassan. 2017. "Effective Heuristics for Ant Colony Optimization to Handle Large-Scale Problems." *Swarm and Evolutionary Computation* 32(March): 140–49.

Jamil, Momin, and Xin-She Yang. 2013. "A Literature Survey of Benchmark Functions For Global Optimization Problems Citation Details: Momin Jamil and Xin-She Yang, A Literature Survey of Benchmark Functions for Global Optimization Problems." *Int. Journal of Mathematical Modelling and Numerical Optimisation* 4(2): 150–94.

Jana, N.D., J. Sil, and S. Das. 2015. "Improved Bees Algorithm for Protein Structure Prediction Using AB Off-Lattice Model." In *Advances in Intelligent Systems and Computing*, ed. Matoušek R. (eds) Mendel 2015. Springer, Cham, 39–52.

Javidy, Behzad, Abdolreza Hatamlou, and Seyedali Mirjalili. 2015. "Ions Motion Algorithm for Solving Optimization Problems." *Applied Soft Computing Journal* 32: 72–79.

Jiang, Lian Lian, Douglas L. Maskell, and Jagdish C. Patra. 2013. “A Novel Ant Colony Optimization-Based Maximum Power Point Tracking for Photovoltaic Systems under Partially Shaded Conditions.” *Energy and Buildings* 58: 227–36.

Jones, Karl O, and André Bouffet. 2008. “COMPARISON OF BEES ALGORITHM, ANT COLONY OPTIMISATION AND PARTICLE SWARM OPTIMISATION FOR PID CONTROLLER TUNING.” In *CompSysTech ’08 Proceedings of the 9th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing*, Gabrovo, Bulgaria.

Kannan, B.K., and S.N. Kramer. 1994. “An Augmented Lagrange Multiplier Based Method for Mixed Integer Discrete Continuous Optimization and Its Applications to Mechanical Design. Journal of Mechanical Design, 116(2), 405–411..Pdf.” *Journal of Mechanical Design* 116(June 1994): 405–11.

Karaboga, Dervis. 2005. Technical Report-TR06 *An Idea Based on Honey Bee Swarm for Numerical Optimization*. Erciyes.

Karaboga, Dervis, and Bahriye Akay. 2009. “A Comparative Study of Artificial Bee Colony Algorithm.” *Applied Mathematics and Computation* 214(1): 108–32.

Karaboga, Dervis, and Beyza Gorkemli. 2012. “A Quick Artificial Bee Colony -QABC- Algorithm for Optimization Problems.” In *International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*, Trabzon: IEEE, 1–5.

Karaboga, Dervis, Beyza Gorkemli, Celal Ozturk, and Nurhan Karaboga. 2014. “A Comprehensive Survey: Artificial Bee Colony (ABC) Algorithm and Applications.” *Artificial Intelligence Review* 42(1): 21–57.

Kaveh, A. 2017. “Optimum Design of Multi-Span Composite Box Girder Bridges Using Cuckoo Search Algorithm.” In *Applications of Metaheuristic Optimization Algorithms in Civil Engineering*, Springer International Publishing, 31–46.

Kennedy, J, and R Eberhart. 1995. “Particle Swarm Optimization.” *IEEE International Conference on Neural Networks, 1995. Proceedings.* 4: 1942–48.

Khoshgoftar Manesh, M. H., and M. Ameryan. 2016. “Optimal Design of a Solar-Hybrid Cogeneration Cycle Using Cuckoo Search Algorithm.” *Applied Thermal Engineering* 102: 1300–1313.

Kiran, Mustafa Servet. 2015. “TSA: Tree-Seed Algorithm for Continuous Optimization.” *Expert Systems with Applications* 42(19): 6686–98.

Kiran, Mustafa Servet, Huseyin Hakli, Mesut Gunduz, and Harun Uguz. 2015. “Artificial Bee Colony Algorithm with Variable Search Strategy for Continuous Optimization.” *Information Sciences* 300(1): 140–57.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi. 1983. “Optimization by Simulated Annealing.” *Science* 220(4598): 671–80.

Koupaei, J. Alikhani, S.M.M. Hosseini, and F.M. Maalek Ghaini. 2016. “A New Optimization Algorithm Based on Chaotic Maps and Golden Section Search Method.” *Engineering Applications of Artificial Intelligence* 50: 201–14.

Krishnanand, K. N., and D. Ghose. 2005. “Detection of Multiple Source Locations Using a Glowworm Metaphor with Applications to Collective Robotics.” *Proceedings - 2005*

IEEE Swarm Intelligence Symposium, SIS 2005 2005: 87–94.

Lara, Carlos, Juan J Flores, and Felix Calderon. 2008. "Solving a School Timetabling Problem Using a Bee Algorithm." In *MICAI 2008: Advances in Artificial Intelligence, 7th Mexican International Conference on Artificial Intelligence,* Atizapán de Zaragoza, Mexico.

Larrañaga, Pedro, and Jose A Lozano. 2002. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer, Boston, MA.

Lee, Ji Young, and Ahmed Haj Darwish. 2008. "Multi-Objective Environmental/Economic Dispatch Using the Bees Algorithm with Weighted Sum." In *EKC2008 Proceedings of the EU-Korea Conference on Science and Technology.*, Springer Berlin Heidelberg.

Lewis, Andrew. 2014. "Biogeography-Based Optimisation with Chaos."

Li, Bing, and Weisun Jiang. 1998. "Optimizing Complex Functions by Chaos Search." *Cybernetics and Systems* 29(4): 409–19.

Li, Huanzhe, Kunqi Liu, and Ning Li. 2010. "Improved Bees Algorithm for the Large-Scale Layout Optimization without Performance Constraints." *2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)* 2: 459–63.

Li, Xianneng, and Guangfei Yang. 2016. "Artificial Bee Colony Algorithm with Memory." *Applied Soft Computing Journal* 41: 362–72.

Liao, T. Warren, and Poan Su. 2017. "Parallel Machine Scheduling in Fuzzy Environment with Hybrid Ant Colony Optimization Including a Comparison of Fuzzy Number Ranking Methods in Consideration of Spread of Fuzziness." *Applied Soft Computing Journal* 56: 65–81.

Lien, Li Chuan, and Min Yuan Cheng. 2014. "Particle Bee Algorithm for Tower Crane Layout with Material Quantity Supply and Demand Optimization." *Automation in Construction* 45: 25–32.

Liu, Bo et al. 2005. "Improved Particle Swarm Optimization Combined with Chaos." *Chaos, Solitons & Fractals* 25(5): 1261–71.

Liu, Hui, Zixing Cai, and Yong Wang. 2010. "Hybridizing Particle Swarm Optimization with Differential Evolution for Constrained Numerical and Engineering Optimization." *Applied Soft Computing Journal* 10(2): 629–40.

Liu, Xiaoyong, and Hui Fu. 2014. "PSO-Based Support Vector Machine with Cuckoo Search Technique for Clinical Disease Diagnoses." *The Scientific World Journal* 2014: 1–7.

Loi Lei Lai, and Tze Fun Chan. 2007. *Distributed Generation: Induction and Permanent Magnet Generators*. ed. IEEE Press. John Wiley & Sons Ltd.

Lorenz, E. N. 1963. "Determinitstic Nonperiodic Flow." *Journal of the Atmospheric Sciences* 20: 130–41.

Lu, Xueyan, and Yongquan Zhou. 2008. "A Novel Global Convergence Algorithm: Bee Collecting Pollen Algorithm." *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5227 LNAI: 518–25.

Ma, Shifa, Xia Li, and Yumei Cai. 2017. "Delimiting the Urban Growth Boundaries with a

Modified Ant Colony Optimization Model." *Computers, Environment and Urban Systems* 62: 146–55.

Mahmuddin, M., and Y. Yusof. 2010. "Automatic Estimation Total Number of Cluster Using a Hybrid Test-and-Generate and K-Means Algorithm." In *2010 International Conference on Computer Applications and Industrial Electronics*, Kuala Lumpur, Malaysia, 593–96.

Majumder, Irani. 2016. "Controller for Improving Stability of a PV Based Micro Grid." In *2016 International Conference on Circuit, Power and Computing Technologies [ICCPCT]*, IEEE, 1–5.

Maleki, Akbar. 2017. "Design and Optimization of Autonomous Solar-Wind-Reverse Osmosis Desalination Systems Coupling Battery and Hydrogen Energy Storage by an Improved Bee Algorithm." *Desalination*.

Martino, Giada, Baris Yuce, Raffaele Iannone, and Michael S. Packianather. 2016. "Optimisation of the Replenishment Problem in the Fashion Retail Industry Using Tabu-Bees Algorithm." *IFAC-PapersOnLine* 49(12): 1685–90.

Mavrovouniotis, Michalis, Changhe Li, and Shengxiang Yang. 2017. "A Survey of Swarm Intelligence for Dynamic Optimization: Algorithms and Applications." *Swarm and Evolutionary Computation* 33(December 2016): 1–17.

Mehdinejadiani, Behrouz. 2017. "Estimating the Solute Transport Parameters of the Spatial Fractional Advection-Dispersion Equation Using Bees Algorithm." *Journal of Contaminant Hydrology* 203(November 2016): 51–61.

Mehrabian, A. R., and C. Lucas. 2006. "A Novel Numerical Optimization Algorithm Inspired from Weed Colonization." *Ecological Informatics* 1(4): 355–66.

Mezura-Montes, Efrén, and Carlos A. Coello Coello. 2005. "Useful Infeasible Solutions in Engineering Optimization with Evolutionary Algorithms." In *MICAI 2005: Advances in Artificial Intelligence*, 652–62.

Mirjalili, Seyedali. 2016. "Dragonfly Algorithm: A New Meta-Heuristic Optimization Technique for Solving Single-Objective, Discrete, and Multi-Objective Problems." *Neural Computing and Applications* 27(4): 1053–73.

Mirjalili, Seyedali. 2017. "Salp Swarm Algorithm: A Bio-Inspired Optimizer for Engineering Design Problems." *Advances in Engineering Software* 114: 163–91.

Mirjalili, Seyedali, and Andrew Lewis. 2016. "The Whale Optimization Algorithm." *Advances in Engineering Software* 95: 51–67.

Mirjalili, Seyedali, Seyed Mohammad Mirjalili, and Andrew Lewis. 2014. "Grey Wolf Optimizer." *Advances in Engineering Software* 69: 46–61.

Mohamad, Azizah, Azlan Mohd Zain, Nor Erne Nazira Bazin, and Amirmudin Udin. 2015. "A Process Prediction Model Based on Cuckoo Algorithm for Abrasive Waterjet Machining." *Journal of Intelligent Manufacturing* 26(6): 1247–52.

Mohamad Idris, R., A. Khairuddin, and M. W. Mustafa. 2009. "A Multi-Objective Bees Algorithm for Optimum Allocation of FACTS Devices for Restructured Power System." In *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, 1–6.

Mohamed, Al Attar Ali, Yahia S. Mohamed, Ahmed A.M. El-Gaafary, and Ashraf M. Hemeida. 2017. “Optimal Power Flow Using Moth Swarm Algorithm.” *Electric Power Systems Research* 142: 190–206.

Moser, Irene, and Raymond Chiong. 2009. “A Hooke-Jeeves Based Memetic Algorithm for Solving Dynamic Optimisation Problems.” *Artificial Intelligence*: 301–9.

Mucherino, Antonio, and Onur Seref. 2007. “Monkey Search: A Novel Metaheuristic Search for Global Optimization.” *AIP Conference Proceedings* 953(1): 162–73.

Muhamad, Zaidi, Massudi Mahmuddin, Mohammad Faidzul Nasrudin, and Shahnorbanun Sahran. 2011. “Local Search Manoeuvres Recruitment in The Bees Algorithm.” In *Proceeding of The 3rd International Conference on Computing and Informatics*, Bandung, Indonesia: University Utara Malaysia Press, 43–48.

Nasrinpour, Hamid, Amir Bavani, and Mohammad Teshnehlab. 2017. “Grouped Bees Algorithm: A Grouped Version of the Bees Algorithm.” *Computers* 6(1): 5.

Nguyen, Duc Hoang. 2015. “A Hybrid SFL-Bees Algorithm.” *International Journal of Computer Applications* 128(5): 13–18.

Nguyen, Thang Trung, and Dieu Ngoc Vo. 2016. “Solving Short-Term Cascaded Hydrothermal Scheduling Problem Using Modified Cuckoo Search Algorithm.” *International Journal of Grid and Distributed Computing* 9(1): 67–78.

Nguyen, Thang Trung, Dieu Ngoc Vo, and Bach Hoang Dinh. 2016. “Cuckoo Search Algorithm Using Different Distributions for Short-Term Hydrothermal Scheduling with Reservoir Volume Constraint.” *International Journal on Electrical Engineering and Informatics* 8(1): 76–92.

Olivas, Frumen et al. 2017. “Ant Colony Optimization with Dynamic Parameter Adaptation Based on Interval Type-2 Fuzzy Logic Systems.” *Applied Soft Computing Journal* 53: 74–87.

Osman, I.H., and Gilbert Laporte. 1996. “Metaheuristics: A Bibliography.” *Annals of Operations Research* 63: 513–623.

Özbakir, Lale, Adil Baykasoğlu, and Pinar Tapkan. 2010. “Bees Algorithm for Generalized Assignment Problem.” *Applied Mathematics and Computation* 215(11): 3782–95.

Packianather, M. S., M. Landy, and D. T. Pham. 2009. “Enhancing the Speed of the Bees Algorithm Using Pheromone-Based Recruitment.” In *IEEE International Conference on Industrial Informatics (INDIN)*.

Packianather, Michael S et al. 2014. “Novel Genetic Bees Algorithm Applied to Single Machine Scheduling Problem.” In *World Automation Congress 2014*, 1–6.

Pan, Wen-Tsao. 2012. “A New Fruit Fly Optimization Algorithm: Taking the Financial Distress Model as an Example.” *Knowledge-Based Systems* 26: 69–74.

Parsopoulos, K E, and M N Vrahatis. 2005. “Unified Particle Swarm Optimization for Solving Constrained Engineering Optimization Problems.” *Advances in natural computation, Springer*: 582–91.

Passino, Kevin M. 2010. “Bacterial Foraging Optimization.” *International Journal of Swarm Intelligence Research* 1(1): 1–16.

Pham, D. T., Anthony J. Soroka, et al. 2006. “Optimising Neural Networks for Identification of Wood Defects Using the Bees Algorithm.” *2006 IEEE International Conference on Industrial Informatics, INDIN’06*: 1346–51.

Pham, D. T., M. Castellani, and A. A. Fahmy. 2008. “Learning the Inverse Kinematics of a Robot Manipulator Using the Bees Algorithm.” In *IEEE International Conference on Industrial Informatics (INDIN 2008)*, Daejeon, Korea, 493–98.

Pham, D.T. et al. 2005. 0501 The Manufacturing Engineering Centre *Bee Algorithm - A Novel Approach to Function Optimisation, Technical Report*. Cardiff, UK.

Pham, D.T., S. Otri, et al. 2007. “Data Clustering Using the Bees Algorithm.” In *40th CIRP International Manufacturing Systems Seminar*.

Pham, D.T., and M. Castellani. 2009. “The Bees Algorithm: Modelling Foraging Behaviour to Solve Continuous Optimization Problems.” *Proceedings of the Institution of Mechanical Engineers* Part C(223): 2919–2938.

Pham, D.T., and Marco Castellani. 2015. “A Comparative Study of the Bees Algorithm as a Tool for Function Optimisation.” *Cogent Engineering* 2(1): 1091540.

Pham, D.T., and Mete Kalyoncu. 2009. “Optimisation of a Fuzzy Logic Controller for a Flexible Single-Link Robot Arm Using the Bees Algorithm.” *7th IEEE International Conference on Industrial Informatics (INDIN 2009)*: 475–80.

Pham, D.T., S. Otri, A. Ghanbarzadeh, and E. Koc. 2006a. “Application of the Bees Algorithm to the Training of Learning Vector Quantisation Networks for Control Chart Pattern Recognition.” In *2nd International Conference on Information & Communication Technologies ICTTA ’06*, IEEE, 1624–29.

Pham, D T, Z Muhamad, et al. 2007. “Using the Bees Algorithm to Optimise a Support Vector Machine for Wood Defect Classification.” In *3rd International Virtual Conference on Intelligent Production Machines and Systems (IPROMS 2007)*.

Pham, D T, H. AL-Jabbouli, et al. 2008. “Application of the Bees Algorithm to Fuzzy Clustering.” In *4th International Conference on Intelligent Production Machines and Systems (IPROMS 2008)*, Scotland.

Pham, D T, A A Afify, and E Koç. 2007. “Manufacturing Cell Formation Using the Bees Algorithm.” *3rd International Virtual Conference on Intelligent Production Machines and Systems (IPROMS 2007)* (August 2015): 523–28.

Pham, D T, M Castellani, and A Ghanbarzadeh. 2007. “Preliminary Design Using the Bees Algorithm.” In *8th LAMDAMAP International Conference on Laser Metrology, CMM and Machine Tool Performance*, Cardiff, UK, 420–29.

Pham, D T, and A. Haj Darwish. 2008. “Fuzzy Selection of Local Search Sites in the Bees Algorithm.” In *4th International Virtual Conference on Intelligent Production Machines and Systems (IPROMS 2008)*.

Pham, D T, and A Darwish. 2010. “Using the Bees Algorithm with Kalman Filtering to Train an Artificial Neural Network for Pattern Classification.” *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering* 224(7): 885–92.

Pham, D T, Ahmed Haj Darwish, Eldaw Elzaki Eldukhri, and Sameh Otri. 2007. “Using the

Bees Algorithm to Tune a Fuzzy Logic Controller for a Robot Gymnast." In *2007 Innovative Production Machines and Systems*.

Pham, D T, A Ghanbarzadeh, S Otri, and E Koç. 2009. "Optimal Design of Mechanical Components Using the Bees Algorithm." *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 223(5): 1051–56.

Pham, D T, and Afshin Ghanbarzadeh. 2007. "Multi-Objective Optimisation Using the Bees Algorithm." In *3rd International Virtual Conference on Intelligent Production Machines and Systems (IPROMS 2007)*, Whittles, Dunbeath, Scotland.

Pham, D T, E Koç, and A Ghanbarzadeh. 2006. "Optimization of the Weights of Multi-Layered Perceptions Using the Bees Algorithm." In *Proceedings of 5th International Symposium on Intelligent Manufacturing Systems*, , 38–46.

Pham, D T, E Koç, J Y Lee, and J Phrueksanant. 2007. "Using the Bees Algorithm to Schedule Jobs for a Machine." In *Eighth International Conference on Laser Metrology, CMM and Machine Tool Performance*, , 430–39.

Pham, D T, J Y Lee, A Haj Darwish, and A J Soroka. 2008. "Multi-Objective Environmental/Economic Power Dispatch Using the Bees Algorithm with Pareto Optimality." In *4th International Virtual Conference on Intelligent Production Machines and Systems (IPROMS 2008)*, Cardiff, UK.

Pham, D T, S Otri, and AH Darwish. 2007. "Application of the Bees Algorithm to PCB Assembly Optimisation." In *3rd Virtual International Conference on Innovative Production Machines and Systems*, Cardiff, UK, 511–516.

Pham, D T, Sameh Otri, A Ghanbarzadeh, and E Koc. 2006b. "Application of the Bees Algorithm to the Training of Radial Basis Function Networks for Control Chart Pattern Recognition." In *5th CIRP International Seminar on Intelligent Computation in Manufacturing Engineering (CIRP ICME'06)*, Ischia, Italy, 711–16.

Pham, Ly Huu, Thang Trung Nguyen, Dieu Ngoc Vo, and Cuong Dinh Tran. 2016. "Adaptive Cuckoo Search Algorithm Based Method for Economic Load Dispatch with Multiple Fuel Options and Valve Point Effect." *International Journal of Hybrid Information Technology* 9(1): 41–50.

Pham, Q.T., D.T. Pham, and M. Castellani. 2012. "A Modified Bees Algorithm and a Statistics-Based Method for Tuning Its Parameters." *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering* 226(3): 287–301.

Qu, Chiwen, and Wei He. 2016. "A Cuckoo Search Algorithm with Complex Local Search Method for Solving Engineering Structural Optimization Problem." *MATEC Web of Conferences* 9.

Rajakumar, B.R. 2012. "The Lion's Algorithm: A New Nature-Inspired Search Algorithm." *Procedia Technology* 6: 126–35.

Rashedi, Esmat, Hossein Nezamabadi-pour, and Saeid Saryazdi. 2009. "GSA: A Gravitational Search Algorithm." *Information Sciences* 179(13): 2232–48.

Ray, T, and K M Liew. 2003. "Society and Civilization: An Optimization Algorithm Based on the Simulation of Social Behavior." *IEEE Transactions on Evolutionary Computation* 7(4): 386–96.

Rezaee Jordehi, Ahmad, and Jasronita Jasni. 2012. “Particle Swarm Optimisation for Discrete Optimisation Problems: A Review.” *Artificial Intelligence Review* 43(2): 243–58.

Sadiq, Ahmed T., and Amaal G Hamad. 2012. “Exploration-Balanced Bees Algorithms to Solve Optimization and NP-Complete Problems.” *International Journal of Research and Reviews in Soft and Intelligent Computing (IJRRSIC)* 2(1): 108–13.

Sanajaoba, Sarangthem, and Eugene Fernandez. 2016. “Maiden Application of Cuckoo Search Algorithm for Optimal Sizing of a Remote Hybrid Renewable Energy System.” *Renewable Energy* 96: 1–10.

Sayadi, Fares, Mahamod Ismail, Norbahiah Misran, and Kasmiran Jumari. 2009. “Multi-Objective Optimization Using the Bees Algorithm in Time-Varying Channel for MIMO MC-CDMA Systems.” *European Journal of Scientific Research* 33(3): 411–28.

Sekhar, Pudi, and Sanjeeb Mohanty. 2016. “An Enhanced Cuckoo Search Algorithm Based Contingency Constrained Economic Load Dispatch for Security Enhancement.” *International Journal of Electrical Power and Energy Systems* 75: 303–10.

Shatnawi, Nahlah, Shahnorbanun Sahran, and Mohammad Faidzul. 2013. “A Memory Based Bees Algorithm: An Enhancement.” *Journal of Applied Sciences* 13(3): 497–502.

Socha, Krzysztof. 2004. “ACO for Continuous and Mixed-Variable Optimization.” *International Workshop on Ant Colony Optimization and Swarm Intelligence (ANTS’04), Brussels, Belgium* LNCS Vol.: 25–36.

Socha, Krzysztof, and Marco Dorigo. 2008. “Ant Colony Optimization for Continuous Domains.” *European Journal of Operational Research* 185(3): 1155–73.

Sreenivasa Rao, M., and N. Venkaiah. 2017. “A Modified Cuckoo Search Algorithm to Optimize Wire-EDM Process While Machining Inconel-690.” *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 39(5): 1647–61.

Sri Madhava Raja, N., and R. Vishnupriya. 2016. “Kapur’s Entropy and Cuckoo Search Algorithm Assisted Segmentation and Analysis of RGB Images.” *Indian Journal of Science and Technology* 9(17).

Storn, Rainer, and Kenneth Price. 1997. “Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces.” *Journal of Global Optimization* 11(4): 341–59.

Sudabattula, Sureshkumar, and M. Kowsalya. 2016. “Optimal Allocation of Wind Based Distributed Generators in Distribution System Using Cuckoo Search Algorithm.” *Procedia Computer Science* 92: 298–304.

Talbi, El Ghazali. 2009. Metaheuristics: From Design to Implementation *Metaheuristics: From Design to Implementation*. Hoboken, NJ, USA: Wiley.

Tan, Y., and Y. Zhu. 2005. “Fireworks Algorithm for Optimization.” In *Advances in Swarm Intelligence. ICSI 2010. Lecture Notes in Computer Science*, eds. Y. Tan, Y. Shi, and K.C. Tan. Springer, Berlin, Heidelberg, 355–64.

Tandis, Emad, and Ehsanolah Assareh. 2017. “Inverse Design of Airfoils via an Intelligent Hybrid Optimization Technique.” *Engineering with Computers* 33(3): 361–74.

Tariq Sadiq, Ahmed, and Amaal Ghazi Hamad. 2010. “BSA: A Hybrid Bees’ Simulated

Annealing Algorithm To Solve Optimization & NP-Complete Problems." *Eng. & Tech. Journal* 28(2): 271–81.

Teodorović, Dušan, Panta Lucic, Goran Markovic, and Mauro Dell'Orco. 2006. "Bee Colony Optimization: Principles and Applications." *8th Seminar on Neural Network Applications in Electrical Engineering (NEUREL 2006)*: 151–56.

Tereshko, Valery, and Andreas Loengarov. 2005. "Collective Decision-Making in Honey Bee Foraging Dynamics." *Computing and Information Systems Journal* 9(3): 1–7.

Tsai, Hsing Chih. 2014. "Novel Bees Algorithm: Stochastic Self-Adaptive Neighborhood." *Applied Mathematics and Computation* 247: 1161–72.

Tsutsui, Shigeyoshi et al. 2001. IlliGAL Report 2001019 *Evolutionary Algorithm Using Marginal Histogram Models in Continuous Domain*.

Tsutsui, Shigeyoshi. 2004. "Ant Colony Optimisation for Continuous Domains with Aggregation Pheromones Metaphor." *International Conference.*: 207–12.

Uymaz, Sait Ali, Gulay Tezel, and Esra Yel. 2015. "Artificial Algae Algorithm (AAA) for Nonlinear Global Optimization." *Applied Soft Computing Journal* 31: 153–71.

Uysal, Fatih et al. 2017. "Estimating Seebeck Coefficient of a P-Type High Temperature Thermoelectric Material Using Bee Algorithm Multi-Layer Perception." *Journal of Electronic Materials* 46(8): 4931–38.

Varadhan, Ravi, and Hans W. Borchers. 2011. "Dfoptim: Derivative-Free Optimization. R Package Version 2011.8-1. Http://CRAN.R-Project.Org/Package=dfoptim."

Vora, Megha, and T. T. Mirnalinee. 2015. "From Optimization to Clustering: A Swarm Intelligence Approach." In *Handbook of Research on Artificial Intelligence Techniques and Algorithms*, ed. Pandian Vasant. Hershey, PA: IGI Global, 594–619.

Weise, Thomas. 2009. 1 *Global Optimization Algorithms–Theory and Application*. 2nd ed.

Xie, Yongquan et al. 2015. "A Forager Adjustment Strategy Used by the Bees Algorithm for Solving Optimization Problems in Cloud Manufacturing." In *ASME 2015 International Manufacturing Science and Engineering Conference. Volume 2: Materials; Biomanufacturing; Properties, Applications and Systems; Sustainable Manufacturing*, Charlotte, North Carolina, USA: ASME, 1–7.

Xu, S. et al. 2011. "Adaptive Bees Algorithm-Bioinspiration from Honeybee Foraging to Optimize Fuel Economy of a Semi-Track Air-Cushion Vehicle." *The Computer Journal* 54(9): 1416–26.

Xu, Wenjun et al. 2016. "Perception Data-Driven Optimization of Manufacturing Equipment Service Scheduling in Sustainable Manufacturing." *Journal of Manufacturing Systems* 41: 86–101.

Xue, Yu, Jiongming Jiang, Binping Zhao, and Tinghuai Ma. 2017. "A Self-Adaptive Artificial Bee Colony Algorithm Based on Global Best for Global Optimization." *Soft Computing*: 1–18.

Yang, Shiqin, Jianjun Jiang, and Guangxing Yan. 2009. "A Dolphin Partner Optimization." *Proceedings of the 2009 WRI Global Congress on Intelligent Systems, GCIS 2009* 1: 124–28.

Yang, Xin-she. 2008. Nature-Inspired Metaheuristic Algorithms *Firefly Algorithm*. Bristol, UK: Luniver Press.

Yang, Xin-She, and Suash Deb. 2009. "Cuckoo Search via Levy Flights." *2009 World Congress on Nature and Biologically Inspired Computing, NABIC 2009 - Proceedings*: 210–14.

Yang, Xin She. 2010. "A New Metaheuristic Bat-Inspired Algorithm." In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010). Studies in Computational Intelligence*, eds. J.R. González et al. Springer, Berlin, Heidelberg, 65–74.

Yu, James J.Q., and Victor O.K. Li. 2015. "A Social Spider Algorithm for Global Optimization." *Applied Soft Computing Journal* 30: 614–27.

Yuce, B. et al. 2017. "Hybrid Genetic Bees Algorithm Applied to Single Machine Scheduling with Earliness and Tardiness Penalties." *Computers and Industrial Engineering* 113: 842–58.

Yuce, B., D. T. Pham, M. S. Packianather, and E. Mastrocinque. 2015. "An Enhancement to the Bees Algorithm with Slope Angle Computation and Hill Climbing Algorithm and Its Applications on Scheduling and Continuous-Type Optimisation Problem." *Production and Manufacturing Research* 3(1): 3–19.

Yuce, Baris et al. 2013. "Honey Bees Inspired Optimization Method: The Bees Algorithm." *Insects* 4(4): 646–62.

Zabil, M. H. Mohamed, K. Z. Zamli, and K. C. Lim. 2018. "Evaluating Bees Algorithm for Sequence-Based T-Way Testing Test Data Generation." *Indian Journal of Science and Technology* 11(4): 1–20.

Zamani, Abbas Ali, Saeed Tavakoli, and Sadegh Etedali. 2017. "Fractional Order PID Control Design for Semi-Active Control of Smart Base-Isolated Structures: A Multi-Objective Cuckoo Search Approach." *ISA Transactions* 67: 222–32.

Zambrano-Bigiarini, Mauricio, Maurice Clerc, and Rodrigo Rojas. 2013. "Standard Particle Swarm Optimisation 2011 at CEC-2013: A Baseline for Future PSO Improvements." *2013 IEEE Congress on Evolutionary Computation, CEC 2013*: 2337–44.

Zarei, Kobra, Morteza Atabati, and Monire Ahmadi. 2017. "Shuffling Cross–validation–bee Algorithm as a New Descriptor Selection Method for Retention Studies of Pesticides in Biopartitioning Micellar Chromatography." *Journal of Environmental Science and Health - Part B Pesticides, Food Contaminants, and Agricultural Wastes* 52(5): 346–52.

Zheng, Zongqing et al. 2017. "Dynamic Modeling of Manufacturing Capability for Robotic Disassembly in Remanufacturing." *Procedia Manufacturing* 10: 15–25.

Zhou, Jianzhong et al. 2017. "A Multi-Objective Multi-Population Ant Colony Optimization for Economic Emission Dispatch Considering Power System Security." *Applied Mathematical Modelling* 45: 684–704.

Zhou, Yifan, Zhisheng Zhang, Tian Ran Lin, and Lin Ma. 2013. "Maintenance Optimisation of a Multi-State Series-Parallel System Considering Economic Dependence and State-Dependent Inspection Intervals." *Reliability Engineering and System Safety* 111: 248–59.

Zhou, Z. D. et al. 2016. "Bees Algorithm for Multimodal Function Optimisation."

Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 230(5): 867–84.

APPENDICES

APPENDIX A - BENCHMARK TEST FUNCTIONS FOR GLOBAL OPTIMISATION

Function	Dim	Equation	Domain	Optimum
f_1 Martin & Gaddy	2	$f_1(x_1,x_2) = (x_1 - x_2)^2 + \left[\frac{(x_1 + x_2 - 10)}{3}\right]^2$	$[-20,20]^D$	0
f_2 Booth	2	$f_2(x_1,x_2) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	$[-10,10]^D$	0
f_3 Goldstein & Price	2	$f_3(x_1,x_2) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \cdot [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	$[-2,2]^D$	3
f_4 Schaffer	2	$f_4(x_1,x_2) = 0.5 + \frac{(sin\sqrt{x_1^2 + x_2^2})^2 - 0.5}{[1.0 - 0.001(x_1^2 + x_2^2)]^2}$	$[-100,100]^D$	0
f_5 Six Hump Camel	2	$f_5(\vec{x}) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$	$[-5,5]^D$	-1.0316
f_6 Michalewicz	5	$f_6(x) = -\sum_{i=1}^{n} \sin(x_i)\left[\sin(\frac{ix_i^2}{\pi})\right]^{2m}$	$[0,\pi]^D$	-4.687
f_7 Hypersphere	10	$f_7(\overrightarrow{x}) = \sum_{i=1}^{10} x_i^2$	$[-5.12,5.12]^D$	0
f_8 Rosenbrock	10	$f_8(\vec{x}) = \sum_{i=1}^{10} 100(x_{i+1} - x_1^2)^2(1 - x_i)^2$	$[-50,50]^D$	0
f_9 Powell	10	$f_9(\vec{x}) = \sum_{i=1}^{\frac{10}{4}} [(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - +10(x_{4i-3} - x_{4i}{}^4$	$[-4,5]^D$	0
f_{10} Axis	10	$f_{15}(\overrightarrow{x}) = \sum_{i=1}^{10} ix_i^2$	$[-5.12,5.12]^D$	0
f_{11} Ackley	10	$f_{10}(\vec{x}) = 20 - 20e^{-0.2\sqrt{\left(\frac{1}{10}\right)\sum_{i=1}^{10} x_i^2}} - e^{-0.2\sqrt{\left(\frac{1}{10}\right)\sum_{i=1}^{10}\cos(2\pi x_i)}} + e$	$[-32,32]^D$	0
f_{12} Griewank	10	$f_{11}(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{10}(x_1 - 100)^2 \prod_{i=0}^{i=10}\cos\left(\frac{x_i - 100}{\sqrt{i+1}}\right) + 1$	$[-600,600]^D$	0
f_{13} Rastrigin	10	$f_{12}(\vec{x}) = \sum_{i=1}^{10}[(x_i)^2 - 10\cos(2\pi x_i) + 10]$	$[-5.12,5.12]^D$	0
f_{14} Zakharov	10	$f_{13}(\vec{x}) = \sum_{i=1}^{10}\frac{1}{\sum_{j=0}^{j=10}[(x_j - a_{ij})^2 + c_j]}$	$[-5,10]^D$	0
f_{15} Styblinski-Tang	10	$f_{14}(x) = \frac{1}{2}\sum_{i=1}^{d}(x_i^4 - 16x_i^2 + 5x_1)$	$[-5,5]^D$	-391.6599

APPENDIX B - BENCHMARK MECHANICAL DESIGN PROBLEM FOR OPTIMISATION

B.1. Three-bar truss problem

$$f(x) = \left(2\sqrt{2}x_1 + x_2\right) \times l$$

subject to:

$$g_1(x) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2x_1^2 + 2x_1x_2}} P - \sigma \leq 0$$

$$g_2(x) = \frac{x_2}{\sqrt{2x_1^2 + 2x_1x_2}} P - \sigma \leq 0$$

$$g_3(x) = \frac{1}{\sqrt{2x_2 + x_1}} P - \sigma \leq 0$$

$$0 \leq x_1 \leq 1 \quad i = 1,2$$

$$l = 100\ cm, P = 2\frac{kN}{cm^2}, \sigma = 2\frac{kN}{cm^2}$$

B.2. Pressure vessel design problem

$$f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$$

subject to:

$$g_1(x) = -x_1 + 0.0193x_3 \leq 0$$

$$g_2(x) = -x_2 + 0.00954x_3 \leq 0$$

$$g_3(x) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1{,}296{,}000 \leq 0$$

$$g_4(x) = x_4 - 240 \leq 0$$

$$0 \leq x_i \leq 100 \quad i = 1,2$$

$$10 \leq x_i \leq 200 \quad i = 3,4$$

B.3. Tension/compression spring design problem

$$f(x) = (N + 2)Dd^2$$

subject to:

$$g_1(x) = 1 - \frac{D^3N}{71{,}785d^4} \leq 0$$

$$g_2(x) = \frac{4D^2 - dD}{12{,}566(Dd^3 - d^4)} + \frac{1}{5{,}108d^2} - 1 \leq 0$$

$$g_3(x) = 1 - \frac{140.45d}{D^2N} \leq 0$$

$$g_4(x) = \frac{D+d}{1.5} - 1 \leq 0$$

$$0.05 \leq x_1 \leq 2\,, 0.25 \leq x_2 \leq 1.3\,, 2 \leq x_3 \leq 15$$

B.4. Welded beam problem

$$f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$$

subject to:

$$g_1(x) = \tau(x) - \tau_{max} \leq 0$$

$$g_2(x) = \sigma(x) - \sigma_{max} \leq 0$$

$$g_3(x) = x_1 - x_4 \leq 0$$

$$g_4(x) = 0.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0$$

$$g_5(x) = 0.125 - x_1 \leq 0$$

$$g_6(x) = \delta(x) - \delta_{max} \leq 0$$

$$g_7(x) = P - P_c(x) \leq 0$$

where,

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}\,, \tau' = \frac{P}{\sqrt{2}x_1x_2}\ ,\ \tau'' = \frac{MR}{J}\ ,M = P\left(L + \frac{x_2}{2}\right)$$

$$R = \sqrt{\frac{x_2^2}{4} + (\frac{x_1+x_3}{2})^2}\ ,\ J = 2\left[\sqrt{2}x_1x_2\left\{\frac{x_2^2}{12} + \left(\frac{x_1+x_3}{2}\right)^2\right\}\right]\ ,\ \sigma(x) = \frac{6PL}{x_4x_3^2}$$

$$\delta(x) = \frac{4PL^3}{Ex_3^3x_4}\ ,\qquad P_c(x) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

$$P = 6,000\ lb\ ,L = 14\ in\ ,E = 30e6\ psi\ ,G = 12e6\ psi\ ,\tau_{max} = 13,600$$

$$\sigma_{max} = 30,000\ psi\ ,\delta_{max} = 0.25\ in$$

B.5. Speed reducer problem

$$f(x) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3)$$

subject to:

$$g_1(x) = \frac{27}{x_1 x_2^2 x_3} - 1 \leq 0$$

$$g_2(x) = \frac{397.5}{x_1 x_2^2 x_3^2} - 1 \leq 0$$

$$g_3(x) = \frac{1.93 x_4^3}{x_2 x_3 x_6^4} - 1 \leq 0$$

$$g_4(x) = \frac{1.93 x_5^3}{x_2 x_3 x_7^4} - 1 \leq 0$$

$$g_5(x) = \frac{\left(\left(\frac{745 x_4}{x_2 x_3}\right)^2 + 1.69 \times 10^6\right)^{1/2}}{110.0 x_6^3} - 1 \leq 0$$

$$g_6(x) = \frac{\left(\left(\frac{745 x_4}{x_2 x_3}\right)^2 + 157.5 \times 10^6\right)^{1/2}}{85.0 x_7^3} - 1 \leq 0$$

$$g_7(x) = \frac{x_2 x_3}{40} - 1 \leq 0$$

$$g_8(x) = \frac{5 x_2}{x_1} - 1 \leq 0$$

$$g_9(x) = \frac{x_1}{12 x_2} - 1 \leq 0$$

$$g_{10}(x) = \frac{1.5 x_6 + 1.9}{x_4} - 1 \leq 0$$

$$g_{11}(x) = \frac{1.1 x_7 + 1.9}{x_5} - 1 \leq 0$$

where,

$2.6 \leq x_1 \leq 3.6$, $0.7 \leq x_2 \leq 0.8$, $17 \leq x_3 \leq 28$, $7.3 \leq x_4 \leq 8.3$, $7.8 \leq x_5 \leq 8.3$,

$2.9 \leq x_6 \leq 3.9$, $5.0 \leq x_7 \leq 5.5$